Automatic Error Recovery during Industrial Assembly Operations based on Human Demonstrations

Von der
Carl-Friedrich-Gauß-Fakultät
der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines Doktoringenieurs (Dr.-Ing.)

genehmigte Dissertation

von
Dipl.-Ing. Arne Muxfeldt
geboren am 11.10.1986
in Eckernförde

Eingereicht am: 15. 06. 2018
Mündliche Prüfung am: 19. 10. 2018
Referent: Prof. Dr. Jochen J. Steil
Korreferent: Prof. Dr. Jürgen Hesselbach

2018

Preface

This monograph originates from my scientific work as employee of the *Institut für Robotik und Prozessinformatik* at the *Technische Universität Braunschweig*, Germany.

Firstly, I would like to thank Professor Wahl for giving me the chance to start working as a research assistant. In addition, I would like to thank him for supervising me, even though he was about to retire. I would also like to thank Professor Haux. During his time as acting director of the institute, he enabled me to continue with my own research without any constrains. An additional special thanks goes to Professor Steil. After taking over the academic chair, he made it possible for me to continue with my research and to write this monograph. It was helpful that he always had time for a conversation when it was needed to discuss critical research aspects. At this point, I would also like to thank Professor Hesselbach for being the second examiner of this work.

During my time as student I was supervised by Daniel Kubus. At the beginning of my time as a research assistant and especially in the time of the chair vacancy he always had time for a critical discussion of ideas. Thank you for supporting me. At this point, I would also like to thank all of my students. The support in carrying out the extensive user studies was especially helpful. I also have to thank all my friends and family, who have always been willing to participate in user studies.

A special person always agreed to be a participant in user studies. She continually gave me support and motivation, even in the difficult times. For this, my very special thanks goes to my wife Vivien Muxfeldt.

Braunschweig, 15.06.2018

Arne Muxfeldt

Zusammenfassung

Ausgehend von einem Szenario, in dem sich Menschen und Roboter einen Arbeitsraum teilen, wird ein System zur automatischen Behandlung von Fehlerzuständen in automatisierten Montageprozessen vorgestellt. Tritt ein Fehler auf, so wird dieser erkannt und klassifiziert. Handelt es sich um einen bisher unbekannten Fehler, so wird der Mensch, welcher dem Roboter am nächsten ist gebeten, eine Fehlerbehandlung durch Interaktion mit dem Roboter durchzuführen. Diese Fehlerbehandlung wird aufgezeichnet, sodass sie bei einem erneuten Auftreten des gleichen Fehlers wieder angewendet werden kann. Ist der aufgetretene Fehler jedoch bereits bekannt, so wird eine dazu passende Fehlerbehandlung ausgewählt und ausgeführt, ohne dass es zu einer Interaktion kommt. Somit sinkt die Interaktionsrate über die Zeit betrachtet und das System lernt immer mehr Fehler eigenständig zu behandeln. Zusätzlich wird vorgestellt, wie verschiedene und aufgezeichnete Fehlerbehandlungen gemäß vorgegebenen Performancemaßen optimiert werden können.

Zur Realisierung eines solchen Systems wird zunächst ein passendes Eingabegerät zur Durchführung der Fehlerbehandlung benötigt. Im Rahmen einer Benutzerstudie mit 31 Teilnehmern werden typische Eingabegeräte, welche in industriellen Roboterarbeitszellen zu finden sind, auf ihre Eignung untersucht. Dabei zeigte sich, dass die besten Ergebnisse unter Verwendung von *Kinesthetic Guidance* (*KG*) erreicht werden konnten. In einer weiteren Benutzerstudie war es das Ziel, die Eigenschaften von *KG* näher zu bestimmen. An dieser Studie haben 78 Teilnehmer unterschiedlicher Altersgruppen und mit unterschiedlichen Hintergründen teilgenommen. Es zeigte sich, dass bei der Verwendung von *KG* keine Korrelation zwischen persönlichen Eigenschaften, wie zum Beispiel dem Alter oder räumlichen Vorstellungsvermögen, und dem Erfolg der Interaktion gefunden werden konnte. Zusätzlich konnte keine Korrelation zwischen dem erfolgreichen Ausführen einer Montage per Hand oder per *KG* gefunden werden. Hingegen konnte gezeigt werden, dass schnelle Lernerfolge die Intuitivität, welche *KG* nachgesagt wird, belegen. Somit lässt sich sagen, dass *KG* von der Allgemeinheit ohne spezielles Vorwissen zur Interaktion mit Robotern einsetzt werden kann. Es ist jedoch zu beachten, dass es Unterschiede bezüglich der Dauer, den wirkenden Kräften sowie den Bewegungsgeschwindigkeiten bei der Ausführung einer Montage gibt. Hierauf basierend lässt es sich nicht sagen, dass bei der Verwendung von *KG* menschliche Strategien angewendet werden.

Zusätzlich wird mit der *Hierarchical Decomposition* (*HD*) ein Ansatz zur abstrakten Beschreibung von Montagevorgängen vorgestellt. Hierbei wird der Montageprozess von einem Experten in Zustände und Bedingungen für den Wechsel zwischen diesen Zuständen unterteilt. Somit eröffnet die *HD* die Möglichkeit eine Beobachtung des Montagevorgangs, eine Fehlererkennung sowie Klassifikation und sogar eine Fehlervorhersage durchzuführen.

Des Weiteren wird eine Strategierepräsentation eingeführt, um demonstrierte Fehlerbehandlungen speichern und wiederverwenden zu können. Eine besondere Eigenschaft der vorgestellten Strategierepräsentation ist, dass eine Strategie immer auf die End-Effektor Pose des Roboters zu dem Zeitpunkt, an welchem der Fehler auftritt, bezogen ist. Somit beschreibt eine Strategie die Bewegungen, welche zur Fehlerbehandlung durchzuführen sind. Durch die so gewonnene Robustheit gegen Rotation oder Translation lässt sich die Menge der nötigen Strategien, welche per Interaktion durch einen Menschen zu demonstrieren sind, deutlich senken. Um Strategien auswählen zu können, werden vier Auswahlkriterien vorgestellt. Dabei ist es möglich, eine Auswahl nur auf Basis eines Kriteriums zu treffen oder alle zu berücksichtigen, in dem eine Multikriterienoptimierung durchgeführt wird. Durch die Einführung eines Verfahrens zur Optimierung von Strategien kann die Systemperformance bezüglich eines vorgegebenen Performancemaßes gesteigert werden.

In einem anschließenden Experiment wird gezeigt, dass der vorgestellte Ansatz zur Fehlerbehandlung erfolgreich angewendet werden kann. Hierfür sind Strategien, welche in den vorherigen Benutzerstudien aufgezeichnet wurden, wiederverwendet worden. Während des Experiments wird besonderes Augenmerk darauf gerichtet, wie die Auswahl einer Strategie die Wahrscheinlichkeit zum erfolgreichen Behandeln eine Fehlersituation beeinflusst. Zusätzlich wird in dem Experiment bestätigt, dass die Systemperformance durch eine Strategieoptimierung deutlich gesteigert werden kann.

Abstract

Based on a scenario where humans and robots share their workspace, a system for automatically error handling during an automated industrial assembly is presented. If an error occurs, it is first detected and then classified. If it is a previously unknown error, the human closest to the robot will be asked to perform error handling by interacting with the robot. This interaction is recorded so that it can be reapplied if the same error occurs again. If the error is already known, an appropriate error handling is selected and applied without any further human interaction required. Thus, the interaction rate decreases over time and the system learns to handle more and more errors independently. In addition, it is presented how different recorded error handlings can be optimized according to given performance criteria.

For this purpose, a suitable input device for performing the error handling is required first. Therefore, a user study with 31 participants was carried out in order to examine the suitability of various input devices. The focus was on typical input devices which can be found in industrial robot work cells. Here, the best interaction results have been achieved by using *Kinesthetic Guidance* (*KG*). In another user study, the goal was to specify the properties of *KG*. 78 participants from different age groups and with various backgrounds took part in this study. It turned out that when using *KG*, no correlation between personal characteristics such as age or spatial sense and the success of the interaction could be found. In addition, no correlation between successfully performing a manual assembly or a *KG* based one could be found. On the other hand, it could be shown that fast learning successes prove the intuitiveness attributed to *KG*. Thus, it can be said that *KG* can be used by a normal human operator without special prior knowledge in robot interaction. However, it should be noted that there are differences between manual assembly and *KG* regarding the duration, acting forces and movement speed during assembly. Hence, it cannot be said that human strategies are used when applying *KG*.

In addition, the *Hierarchical Decomposition* (*HD*) is introduced as the abstract representation of an assembly operation. In this case, an assembly is subdivided into different states at multiple hierarchical levels. This is done by a domain export which also defines conditions for state transition. Thus, the *HD* allows assembly progress monitoring, error detection and classification as well as error prediction.

A strategy presentation is introduced to store and reuse demonstrated error handling interactions. One particular feature of this representation is that a strategy is always related to the robot's end-effector pose at that point of time when an error occurs. Thus, a strategy describes the movements which have been performed for error handling. The strategy's invariance against rotation or translation allows significant reduction in the amount of strategies needed to be demonstrated by a human via interaction. Four selection criteria are introduced in order to decide if a strategy matches an error. Thereby, it is possible to make a selection based on one criterion or to perform a multi-criteria optimization using all available information. By introducing a strategy optimization approach, the overall system performance can be improved.

In a subsequent experiment, it is shown that the presented error handling approach can be successfully applied. For this experiment, strategies which were recorded during two user studies have been reused. Within the experiment, particular attention is paid to how selecting a strategy affects the likelihood of successful error handling. In addition, it is confirmed by the experiment that the overall system performance can be significantly improved by the strategy optimization approach.

Contents

Chapter 1

Introduction

This introduction is structured in such a way that the current development concerning automated assembly is considered first. In doing so, a research gap based on a new trend is identified. This gap serves as the primary motivation for developing a new error handling approach. Then, the need for research in this direction is underlined by summing up the body of related work. Subsequently, an overview of the new approach is given.

1.1. Motivation

Although industrial robot sales are growing, most of the deployed robots are used for traditional applications in industry e.g. welding or painting [79, 60]. However, with the advent of more lightweight and force-controlled robots, new application fields are emerging in particular within automated assembly, e.g. in spatially restricted workspace such as inside a car body frame. While significant challenges still exist in automated task planning regarding such scenarios, it is safe to assume that more such systems will be deployed, especially where robots operate typically alongside or in close cooperation with human co-workers as depicted in Fig. 1.1 [78]. Human-Robot Interaction (*HRI*) can increase the ergonomics of assembly tasks [56]. For example, a robot handles heavy workpiece and a human performs only processing steps. This is an important point because the increase in life expectancy among European populations, also known as the Ageing of Europe, requires adapting industrial production environments to the ageing workforce. In addition, flexibility and short production cycle time, especially in Small and Medium-Sized Enterprises, increases the need for Human-Robot Collaboration (*HRC*) in order to combine the human flexibility and task-specific knowledge with the efficiency of robotic systems [86, 28]. In summary, humans and robots sharing the same workspace is desirable and will continue becoming more commonplace [44].

Figure 1.1.: Exemplary depiction of a workspace shared by a human and robots. This offers the opportunity to cooperate if e.g. a robot is blocked and needs help to recover. This help can be given by the human through Kinesthetic Guidance.

On the one hand, assembly and task planning can be automated for these systems (cf. Sec. 1.2), while facilitating tasks can be performed with high processing speed and consistently high quality. However, automated assembly suffers from the circumstance that errors, caused by e.g. workpiece or pose tolerances, often occur during assembly. With the increasing task complexity, it is not economically viable to avoid all possible assembly errors. This in turn requires the usage of complex and costly error handling routines. On the other hand, shop-floor workers can react to new situations using their innate task-specific knowledge and experience.

However, if human and robotic aspects are considered together, error handling can be considerably simplified. Here, the focus is on a scenario where the workspace is shared by humans and robots. In case that an error occurs during assembly performed by a robot, a human co-worker can interpose immediately and initiate robot recovery as shown in Fig. 1.2. Thereby, a task and situation specific recovery strategy can be applied because

the human co-workers are aware of assembly error and task-specific constraints. The error handling efficiency can be increased if the human has task specific knowledge and experience, in particular through Kinesthetic Guidance (*KG*) [2]. *KG* is a commonly employed method for programming robots using the *Programming by Demonstration* (*PbD*) paradigm. It is widely regarded as an intuitive approach to robot programming, which can be performed by shop-floor workers [58, 31]. Much research in this area has focused on pick-and-place tasks while demanding assembly tasks have so far received less attention. Storing successful recovery strategies allows later reuse on a similar error. This results in a robotic system which is learning from human intervention. If an already known error occurs again, the robot can recover on its own by using a learned strategy. Thus, the rate of required human interventions can be reduced over time. The advantage of this approach is that no programming capabilities are required by the shop-floor workers and that their knowledge and experience can be integrated into a robotic system.

Figure 1.2.: A shop-floor worker applies Kinesthetic Guidance in an assembly scenario.

Before an error handling approach can be developed, the term '*assembly*' needs to be specified. It is possible to look how it is done by other approaches for specifying assembly tasks [23, 49]. There, a *MTM*-Code is used for decomposing assembly tasks into primitive and directly executable actions [72]. It then becomes possible to map these actions to

the manufacturing techniques according to *DIN8593-1* (cf. Fig. 1.3) [29]. Alternatively, different assembly and task planning approaches can be considered (cf. Sec. 1.2). Here, mating relations are used to describe the connections between workpieces [35, 25, 73]. Again, these relations can be mapped to manufacturing techniques according to *DIN8593-1*. Therefore, the term '*assembly*' is always linked to the standard *DIN 8593-1:2003-09* and all considered assembly tasks have been selected w.r.t. to that standard.

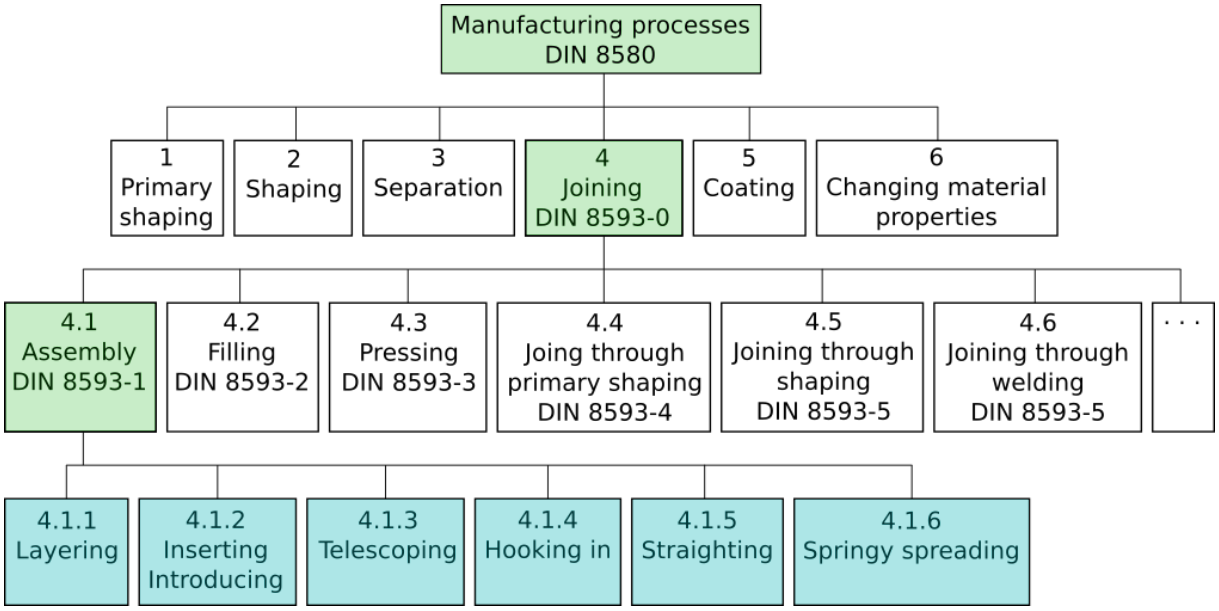


Figure 1.3.: Manufacturing techniques according to DIN8593-1.

1.2. Related Work

In general, there are two classes of approaches for assembly planning. The approaches belonging to the first class are based on CAD models. Their common feature is that assembly and task planning can be automated for these systems [3, 81, 27, 4, 88]. This automation enables the assembly to be performed with high processing speed and consistently high quality. The unique selling point of the second class is that these approaches are based one machine learning [37, 54, 46]. A disadvantage is that it is necessary to learn again for each new task. Also, it can be required that complex assembly tasks are subdivided by an operator before learning is possible.

All mentioned approaches suffer from the fact that error handling is not included. Therefore, extra effort or approaches are needed. But during automated assembly, the process

can fail due to deviations from the model representation, e.g. component tolerances or placement errors. A simple, technically easy to use and therefore frequently used approach is to call for an operator if an error occurs during automated assembly. In this case, it is the operator's task to resolve the error to continue automated operation [16]. But calling an operator requires time, so that in the worst case it is necessary to pause subsequent assembly steps or the whole line. In industry, a standard approach is to use impedance control for handling these errors [76, 20, 70, 13, 12] and for minimizing downtimes. Even so, there is still room for improvement. Thus, an approach was presented which combines impedance control and specific motion patterns for recovering from an error [8]. However, a recent user study suggests that it is rather parameter-sensitive which means not all errors may be handled [57]. Nevertheless, in the same study the human participants were always able to solve the errors even in cases where an impedance controller failed. So, it can be said that task specific knowledge should be considered for recovering from errors. Another more advanced approach uses additional and task-specific knowledge for defining safe points in the flowchart of an assembly process. If an error occurs, the system attempts to return to the last safe state and to start another attempt [47]. This implies the risk of getting trapped in a deadlock if the error recurs on every attempt. In this scenario, true error handling does not take place. The assembly operation can be definitely continued if a handling error approach is used instead of simply retrying. So, more advanced approaches allow performing predefined motion patterns for recovering from an error. These patterns can be either defined based on simple geometric patterns e.g. a spiral, or they can be situation and task specific. But these patterns need to be pre-programmed [15, 63] and only errors which are foreseen at design-time can be optimally handled. This weakness is also shared by approaches for handling specific errors during peg-in-hole tasks [80, 18, 17]. The reason for this is that an error model needs to be developed for each error. A domain and also programming expert is required for creating such error models. This is a time-consuming process. In [50], a more general sensor-based and situation specific approach is proposed. A directional adjustment during assembly is performed based on the measured contact forces. But there is limitation to a specific set of errors. In summary, an approach for error handling in a situation-specific manner is needed. Currently humans still perform better in error handling, which is a strong argument for using their successfully demonstrated strategies for automatic error handling of robotic systems.

Thus, the question needs to be answered how the interaction between humans and robots should be designed. Special attention must be paid to the requirements of industrial assembly. So, several studies have tried to determine which input device or interface is the most appropriate for user intervention. Fellmann et al. [30] tried to find the best input device for controlling continuum robots, and they considered a 3D mouse (space mouse), a 3D haptic input device and a gamepad. A keyboard, a joystick and a camera were used by Mishra et al. [53] for controlling a simulated robotic arm. A Nintendo Wiimote, a virtual keyboard and a gesture based interface on an iPhone, with a social robot have been examined by Rouanet et al. [69]. In the field of teleoperation, a keyboard,

a flystick and a space mouse have been compared by Carmar et al. [74]. None of these user studies laid their focus on the industrial context and typical input devices that are commonly used in practice. Usually, an industrial robot is controlled by using a manual control pendant (*MCP*). Therefore, one possible approach is to design an intuitive *MCP* interface [6]. But this approach would require updating the robot controls of already deployed solutions. That is why the focus should be on input devices like a keyboard and a space mouse for teaching end-effector poses which are readily available by using typical *MCPs*. Also, the robot itself can be used as an input device by applying Kinesthetic Guidance (*KG*) which is most often used in the Programming-by-Demonstration paradigm (*PbD*) [1, 31]. Furthermore, recent research in the field of *HRI* points out that adaptation of the compliance parameters of the robot can increase the quality of *HRI* [22, 67, 33].

PbD is widely used, especially for programming robots with redundant kinematics [83, 42]. Argall et al. composed a survey of various methods in *PbD* [5]. *KG* especially has been used for many purposes. Kormushev et al. used *KG* to teach different positions and an additional haptic device to teach forces [42]. In contrast, Delson et al. used the taught positions and forces to automatically generate robot programs. Their approach was to remove irrelevant parts of the taught trajectory by checking whether each part is within a specified range of acceptable forces and trajectories [21]. More frequently than in industrial robotics, *PbD* is used in the domain of humanoid robots because it seems logical and intuitive to transfer human motions to humanoid robots. Billard et al. used *KG* to speed up the learning process of a humanoid. Furthermore, guidance is regarded as a user-friendly way of *HRI* [10]. Schou et al. combined *KG* and industrial robotics. They used *KG* in combination with task level programming to create a helpful tool for production floor operators [75]. Wrede et al. performed a study to compare a form of assisted *KG* to an unassisted case for a redundant, industrial robot. They also introduced a new human-robot-interface based on *KG* and machine learning [87]. Pais et al. presented another user study in the context of *PbD*. Their focus was to evaluate the user-friendliness of a tactile user interface, but they did not consider the influence of their interface on the interaction [65]. The differences between manual assembly and using *KG* have not been considered and still needs to be implemented. In addition, it needs to be checked how well *KG* can be applied w.r.t. assembly and how well it can be learned by different groups of the population.

Before an error handling can be performed first, a misreading and classification must be made. Therefore, an assembly task representation which can also represent task specific knowledge is required. This industrial need is addressed by various other approaches. Kormushev et al. utilized *PbD* in teaching and merely repeating single skills for handling tasks like opening a door [42]. A hierarchical task network containing multiple skills with pre- and post-conditions is used by Kaelbing et al. [39] for task planning. In contrast to the latter approach, Madsen et al. used *KG* for teaching multiple skills regarding more complex industrial tasks [66]. This approach also groups different skills and their condi-

tions in a hierarchical structure. The present framework is not limited to task specification and task execution is also possible. Another approach is the so called *Manipulation Task* concept by Weidauer et al. [84]. It is based on a hierarchical network containing different user defined skills for specifying and executing various tasks. However, the mentioned approaches have the commonality of being top-down approaches designed for task specification and execution but not for representation of human demonstrations. Typically, the amount of task execution sequences and the number of taught skills is limited if a task is specified. Thus, these approaches are limited to sequences and skills which are known at specification time. It is not ensured that other sequences or skills can be represented even if they belong to the same task. In contrast to that, a bottom-up approach is based on observed task executions. Therefore, each contained skill and sequence is covered by such an approach. There is no restriction to skills or sequences which are established during task specification. Hence, a bottom-up approach should be preferred for representing an assembly operation. There are two different options for how such a representation can be created. The first one is based on supervised segmentation of the assembly operation. Here, a domain expert is doing the time-consuming segmentation [19, 52]. However, task specific knowledge can be considered by the domain expert. The overall effort can be minimized by performing an unsupervised segmentation [48, 7, 68]. Here, there is the disadvantage that a representation is not human-understandable and no task specific knowledge is encoded. However, the named approaches share the restriction that they do not enable a recovery from assembly errors. For this purpose, additional information regarding possible errors is required.

1.3. Error Handling Approach

Automated assembly can fail due to deviations from the model representation, e.g. component tolerances or placement errors. Therefore, it is necessary that the assembly operation is monitored. In case that an error occurs it can then be classified. This is necessary because an error can have different causes and the recovery is cause depended. Therefore, one database containing error specific recovery strategies is used for each error cause. After classifying the error, it is checked in the associated database whether a matching strategy is known. If such a recovery strategy exists it is selected and applied for recovering from the error. Otherwise, a human operator is called for demonstrating a recovery. Doing this, the task specific knowledge and experience of the operator is exploited. The demonstration is recorded and added to the database. From here, it is considered as a recovery strategy. Due to the fact that human demonstrations are rarely optimal a strategy optimization is performed. An important point is that not only a single strategy is optimized. Instead, all strategies inside the database can be improved by the newly added one. The introduced error handling approach is visualized as a flowchart in Fig. 1.4. The strategies and therefore the databases are depended on the assembly task and are independent of the human

operator or the used robot.

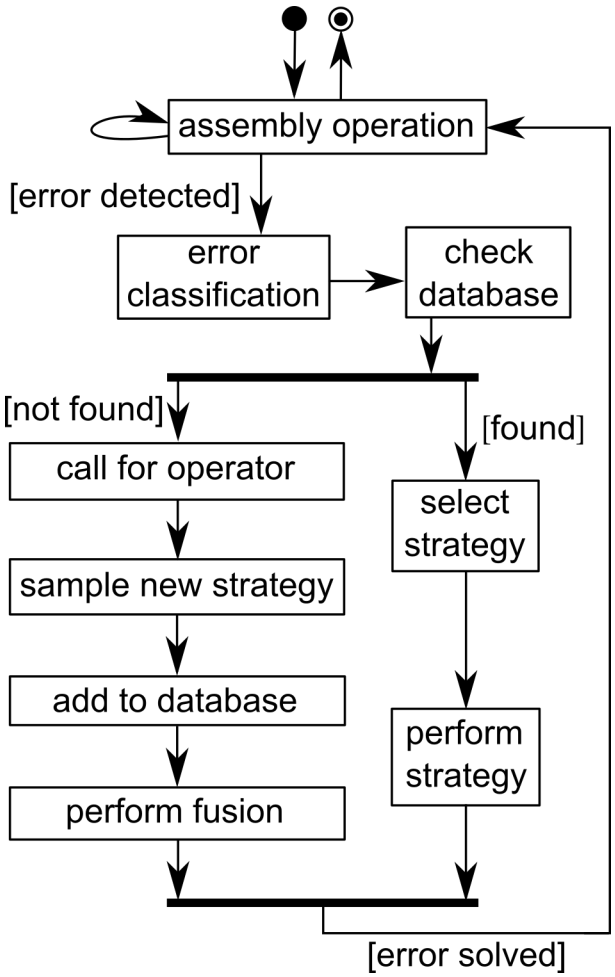


Figure 1.4.: In case that an error occurs during an assembly operation the error gets classified. Afterwards it is checked if a matching recovery strategy can be found in a database. If there is one, the strategy is used for recovering from the error. Otherwise, a human operator is called for demonstrating a new one which is also stored inside the database. For generating additional and optimized recovery strategies the new strategy is fused with existing ones. Afterwards the assembly operation can be continued.

After deploying this error handling approach, it is first necessary to demonstrate recoveries by a human operator. But over time, the number of unknown errors will decrease. Likewise, the number of needed demonstrations is dropping. In the end, an automatic error handling based on the recorded human demonstrations is achieved (cf. Fig. 1.5).

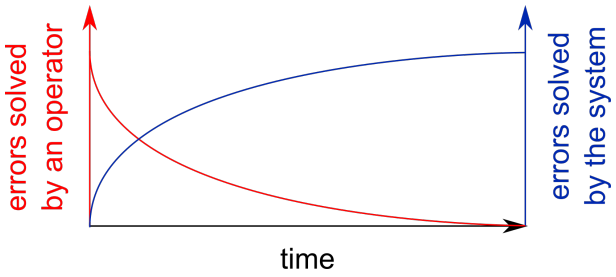


Figure 1.5.: Exemplary plot of how often a human operator needs to interact with the robot for teaching recovery strategies. This is equivalent to the number of errors which cannot be handled by the system on its own.

Before this approach can be deployed as a system, various questions need to answered and components have to be developed. First, an efficient input device for *HRI* w.r.t. assembly scenarios is needed. This question is answered in Chap. 2 by a user study. Subsequently, the characteristics and the general applicability of that input device have to be examined. Therefore, another user study has been carried out and the results are presented in Chap. 3. The Hierarchical Decomposition (*HD*) is introduced in Chap. 4 as a formal and abstract assembly task representation. In addition, the application of the *HD* for error monitoring, detection and classification is explained. In order to record human demonstrations, a strategy representation is presented in Chap. 5. Also, it is show how a strategy can be selected out of a database and how strategies can be optimized. Finally, it is checked in large-scale experiments whether the presented approach is working. The experimental results are given by Chap. 6 and Chap. 7 contains a final conclusion.

Chapter 2

Input Devices for Human-Robot Interaction in Assembly Scenarios

This chapter focuses on Human-Robot Interaction (*HRI*) during assembly. It is based on an earlier publication [57]. A scenario where a shop-floor worker collaborates with a compliant robot is considered here (cf. Fig. 1.2). If an error arises during task execution, a human co-worker can intervene. Since the human is aware of the assembly error and task specific constraints, an initiate and situation-specific recovery strategy can be applied. Now, the question arises which interaction method is the most successful w.r.t. such a scenario. This question is answered by a user study with 31 participants. Within this user study four different interaction methods are compared: Space mouse, Keyboard, Kinesthetic Guidance and Kinesthetic Guidance with adaptive stiffness.

No user study on *HRI* in the field of recovering from assembly errors regarding input devices, which are typically used in the industry, has been performed. Therefore, this study answers which of these input devices should be used if a human co-worker interacts with a robot for recovering the robot from an assembly error.

2.1. Experimental Setup

The experimental setup, shown in Fig. 2.1, consists of a KUKA Light Weight Robot (LWR IV+), a shaft with two mutually displaced keys on it and a conrod with a notch. The conrod is attached to a robot's end-effector. Engineering drawings of the shaft and also of the conrod can be found in Appendix A.6. The shaft is placed on top of a force-torque sensor and the measured forces are visualized on a screen. Using a dedicated force-torque

sensor allows measuring forces even if the robot is not used. This aspect makes it possible to observe manual assembly operations within a preliminary study (cf. Sec. 2.1.2).

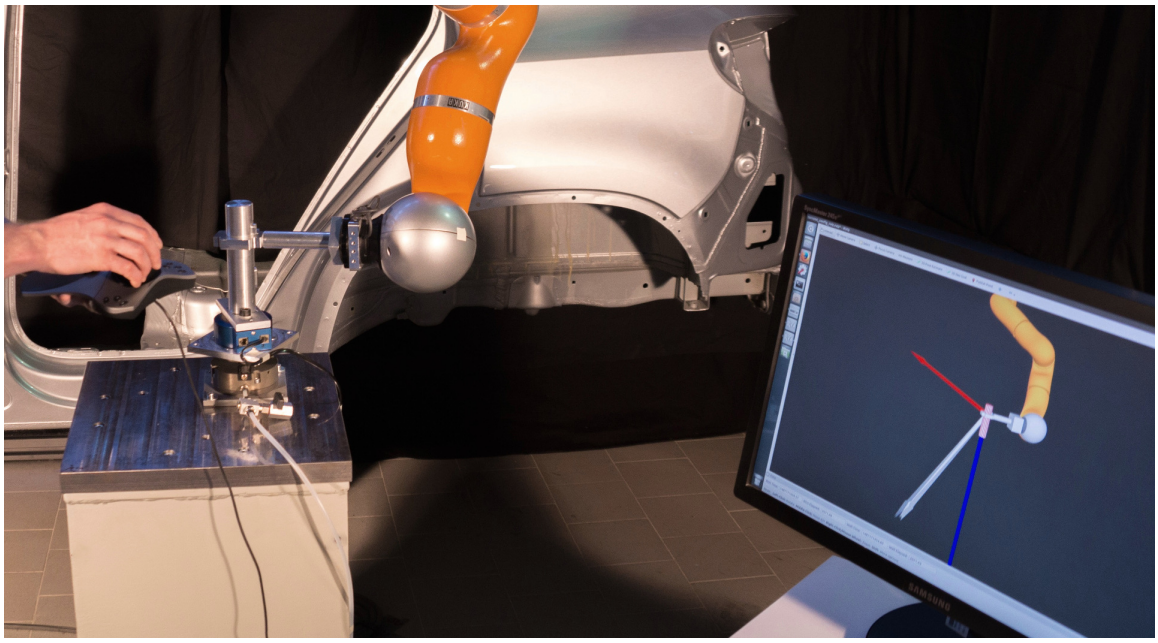


Figure 2.1.: Experimental setup of the user study. A shaft is placed on top of a force-torque sensor and measured forces are visualized on a screen. The robot is attached to the ceiling. A participant who is interacting with the robot by using space mouse is visible.

One of the most important manufacturing techniques considering two workpieces is assembly by mating as specified in *DIN 8593-1:2003-09*. Therefore, the main task of this user study is to mate both workpieces in the mating direction shown in Fig. 2.2, so that the conrod is placed at the lower end of the shaft. An augmented interface to facilitate manipulation of the robot's end-effector (cf. Fig. 2.1) is also part of the experimental setup. This augmented interface visualizes the force vector, its components and the commanded robot motion. Due to the fact that a force-torque sensor is placed below the shaft acting contact forces can be visualized. This enables the participant to compensate any unintentional forces and motions just by observing the force vector performing corresponding inputs.

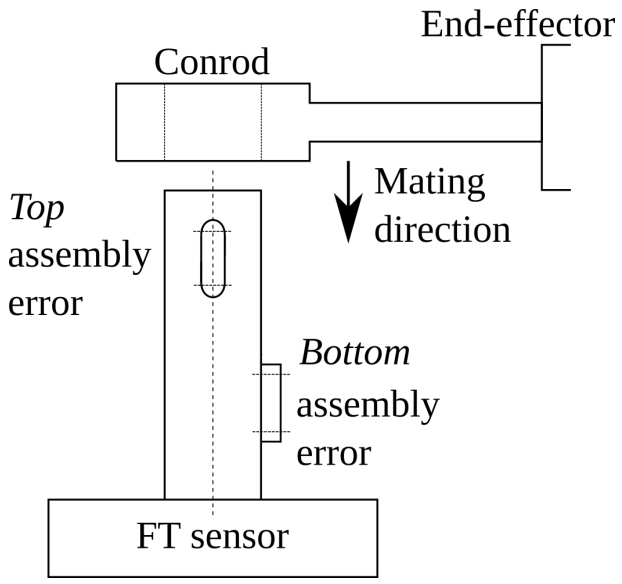


Figure 2.2.: Sketch of the experimental setup for the assembly task *Pleuel*. The workpieces are drawn in the initial position, the joining direction is marked and the assembly error positions are labelled.

2.1.1. Study Design

Overall 31 participants with technical and non-technical background took part in this user study. 22 were male and 9 were female. Altogether 1240 interactions between a human and a robot were observed. Data from all interactions is reused in Chap. 6 for evaluating the developed automatic recovery approach. The presented user study titled M-2017-02 has been approved by the ethics committee of the Technische Universität Braunschweig.

This user study employed a within-subjects design to compare different interaction methods which are mentioned in Sec. 2.2. Here, each participant experienced all four interaction methods while performing two mating tasks. The participants had to recover from each of the two assembly errors using the different methods. Each method was used five times in a row. Both assembly errors have been reliably reproduced as described in Sec. 2.1.2. The *Top* assembly error (cf. Fig. 2.2) is solved first. Then, it is followed by the *Bottom* assembly error with the same interaction method. For each participant, the sequence in which the interaction methods are activated, was randomized to eliminate sequencing effects like learning and tiring.

At the beginning, a video was shown to prevent biasing of the participants. The video described the task, the input devices and the experimental flow. Initially, the robot was in Cartesian impedance mode. It was moved to the initial position where the *Top* assembly

error is created. Then the participant had to recover from an error by using one of the four interaction methods. It was activated randomly and each method involved 5 trials. Once the participant finished the *Top* assembly error, the robot moved to the second position where the *Bottom* assembly error precipitated. Now it was up to the participant to recover from this error in the same manner. Force-torque data and the poses of the robot's end-effector were sampled during each interaction and were used for later analysis.

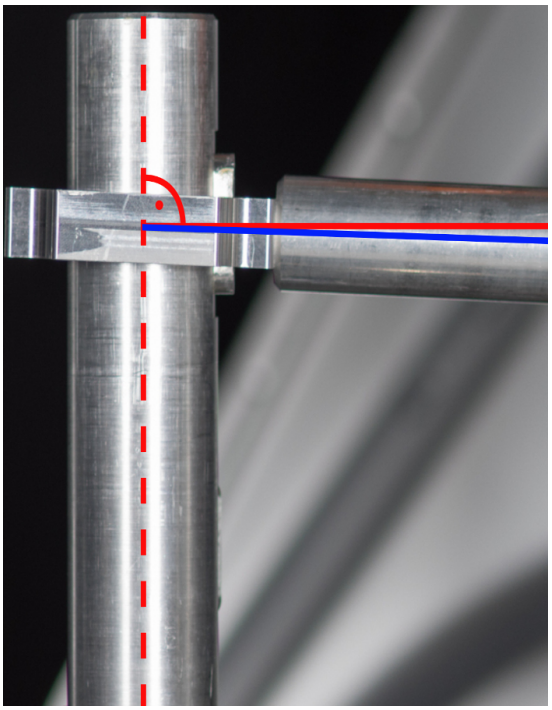
During all trials the participants were free to walk around the setup, making it possible for each participant to choose that field of view which was optimal w.r.t. the interaction method and the considered assembly error. Directly after completing all experiments, the participants were asked to complete a questionnaire where each participant ranked the interaction methods and gave general feedback.

2.1.2. Task Description

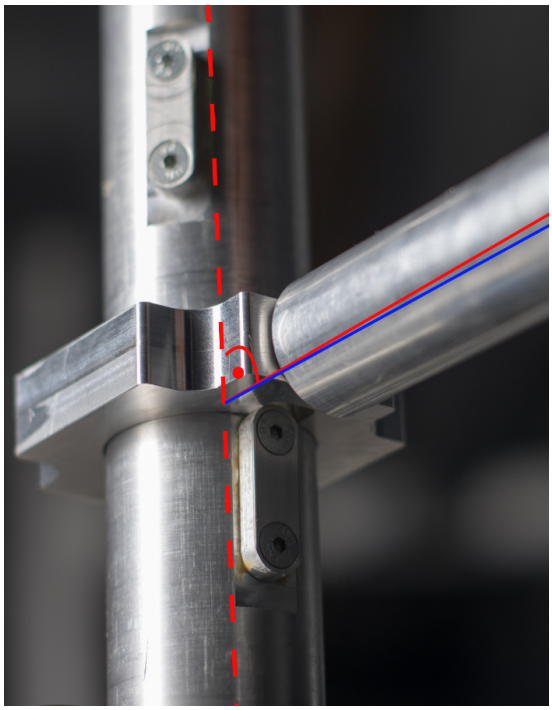
As already mentioned, there are two keys of the shaft (cf. Fig. 2.2). Within this study, only one typical error per key is reproduced. These errors are the following:

- *Top* assembly error: The first error (cf. Fig. 2.3(a)), also called *Top* error, occurs when the conrod is moved along the upper key of the shaft. Thereby a tilting of the conrod against the plane, which is orthogonal to the longitudinal axis of the shaft, occurs.

- *Bottom* assembly error: The second error (cf. Fig. 2.3(b)), also called *Bottom* error, is a rotational incorrect alignment of the conrod which is located above the second key.



(a) *Top* assembly error: The longitudinal axis of the shaft is given by the dashed red line. The blue line is the conrod's axis if there is an error and the red one shows how it should look like for error free assembly.



(b) *Bottom* assembly error: The dashed red line is the longitudinal axis of the shaft. The red line shows how the conrod's axis should be oriented and the blue line exemplary shows the orientation of the conrod during an assembly error.

Figure 2.3.: Overview of the considered assembly errors.

These two errors were not chosen freely but are based on preliminary tests. Therefore, it was possible to determine the complexity and to define meaningful parameters for both errors. Four participants performed the complete assembly task five times. By using an optical tracking system and also the measured forces, it was possible to identify assembly errors during task execution. Here, an error is defined in the way that an assembly operation has stopped and additional actions or inputs are needed to be performed for continuing with the assembly. On average, there were 2.07 errors per task execution.

During the preliminary tests, certain characteristics of the assembly errors were observed. Thereby, errors similar to *Top* and *Bottom* were dominating, but they occurred at both keys. For this reason, the characteristics of these errors are presented now in detail:

- *Top:* The angle between the force vector and the longitudinal axis of the shaft is approximately 85° during mating. This angle decreases to roughly 70° at the beginning of an error.

- *Bottom:* The angle between the force vector and the longitudinal axis of the shaft is equal to approximately 85° during mating. But at the beginning of this error, this angle decreases to roughly 5°.

It should be noted that these errors occur during manual assembly. However, it was also possible to observe such errors in fully automated assembly. So, these errors are of particular importance as they could not be solved in any case by using the KUKA Cartesian impedance controller. Various parameter sets showed only little or no success. Details regarding these experiments can be found in [57].

In addition, it was noticeable that the *Bottom* error was much easier to handle for a human. Only one degree of freedom (*DoF*) had to be manipulated for compensating the rotational misalignment. The *Top* error however was difficult, since a rotational and also a translational *DoF* had to be manipulated. This observation w.r.t. the task complexity is also reflected in the evaluation presented in Sec. 2.3.

2.2. Interaction Methods

Most manual control pendants (*MCP*) feature a keyboard and a space mouse. These two input devices are considered as interaction methods. They can be used without additional cost for deploying new hardware. It is also possible to use the robot itself as an input device by applying *KG*. Beside the classic *KG* approach also a modified version with adaptive stiffness is considered.

2.2.1. Keyboard

A standard keyboard (cf. Sec. 2.1) is used to facilitate intuitive control and to interact with the robot by observing the visualized forces. Specific keys on the keyboard were designated to update the robot position and orientation w.r.t. an individual axis. If a key is pressed during a control cycle, constant displacements c_{dis} are added to the current pose incrementally. It is calculated by multiplying a displacement value d_v with the control cycle time t_{cycle}. Here, d_v was set to 0.5 $\frac{m}{s}$ based on previous experiments.

2.2.2. Space Mouse

Another commonly used device for *HRI* is the space mouse [30, 74]. Here, the participant can only control one *DoF* at a time by moving the space mouse in the desired direction. This simplification makes the interaction more intuitive for untrained and naive participants. During the preliminary tests, an increase of the participants' performance was observable while this simplification was used.

When participants interact with the robot by using the space mouse, the interaction amplitude $amp \in [0, 1]$ is measured, treated as a velocity (translation: $\left[\frac{m}{s}\right]$, rotation: $\left[\frac{rad}{s}\right]$), and scaled. Due to the sensor noise, a threshold of 0.1 $\frac{m}{s}$ is used before the measured amplitude is processed. The commanded end-effector displacement s_{dis} is calculated based on the control cycle time t_{cycle} by:

$$s_{dis} = amp \cdot t_{cycle} \tag{2.1}$$

2.2.3. Kinesthetic Guidance

KG is the most popular approach following the *PbD* paradigm [58], where a human physically guides a robot to perform the desirable skill or the task under consideration [1]. This interaction method is supposed to be highly intuitive (cf. Chap. 3). Hence, accomplishing a task using *KG* will supposedly take very little time even for naive participants or beginners. The accurate modelling of intricate constraints in the task is difficult, but *KG* allows tapping into the implicit knowledge human worker posses about the task constraints.

Here, two *KG* modes are compared. For the first mode, the built-in Cartesian impedance mode of the KUKA robot controller is used. Because no modifications have been done, this mode is called *KG*. For the second mode, the robot's Cartesian stiffness is adapted to the forces, which are applied by the participant. Therefore, this mode is called *Adaptive KG*. The adaptation concept is exemplary depicted in Fig. 2.4. More detailed information regarding this mode can be found in [33]. *Adaptive KG* enables a smoother *HRI*. The underlying assumption is that when the desired Cartesian force in a certain *DoF* is high, then the stiffness value in that direction should be decreased, and vice-versa. Here, the contact force during *HRI* is measured by a force-torque sensor (cf. Sec. 2.1). The force is used together with a linear heuristic to map the acting contact force f_{res} into the stiffness domain, where the $f_{min} = 5\ N$ and $f_{max} = 30\ N$ are the interaction force limits. The stiffness values are limited by $k_{min} = 100\ \frac{N \cdot s}{m}$ and $k_{max} = 1000\ \frac{N \cdot s}{m}$. These values are based on the preliminary experiments.

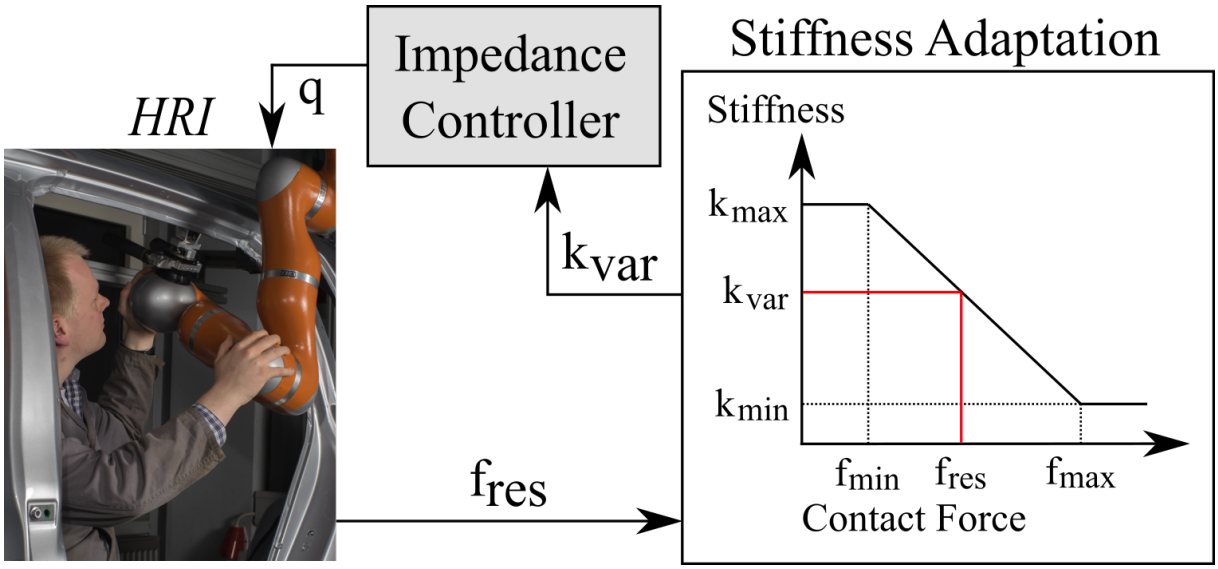


Figure 2.4.: Within each control cycle, a stiffness adaptation is performed and the updated k_{var} is used by the robot's impedance controller. The contact force of the *HRI* is used w.r.t. to the force limits f_{min} and f_{max} and to the stiffness limits k_{min} and k_{max} during the adaptation process.

2.3. Evaluation

The evaluation is based on Tab. 2.1 and Tab. 2.2. These tables have been generated by calculating the performance criteria (cf. Sec. 2.3.1) for the measured data. Some outliers were noticeable during data processing. For determining these suspected outliers as real ones, a Grubbs-test was performed [34]. On average there were 3 (1.9 %) to 6 (3.8 %) outliers per interaction mode and error. They have been removed before Tab. 2.1 and Tab. 2.2 were generated.

Table 2.1.: *Top*: Mean and standard deviations of various performance criteria for each control mode.

	Keyboard		*Space Mouse*		*KG*		*Adaptive KG*	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Jerk (Force) $\left[\frac{N}{s^3}\right]$	12847	11326	14197	13374	1330	979	960	618
Jerk (Pose) $\left[\frac{mm}{s^3}\right]$	45.61	34.19	52.85	45.05	7.99	5.42	6.39	3.78
Arc Length $[mm]$	53.62	21.07	65.00	30.22	40.72	13.40	37.34	16.34
Time $[s]$	60.38	50.72	67.62	64.44	5.96	5.37	4.14	2.57
Effort $\left[N'\right]$	54.99	49.74	100.00	90.94	19.98	15.23	7.30	3.73
Speed $\left[\frac{mm}{s}\right]$	1.04	0.78	1.27	0.93	8.46	5.70	8.80	5.21

Table 2.2.: *Bottom*: Mean and standard deviations of various performance criteria for each control mode.

	Keyboard		*Space Mouse*		*KG*		*Adaptive KG*	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Jerk (Force) $\left[\frac{N}{s^3}\right]$	7007	5206	7718	72898	1506	947	1725	1118
Jerk (Pose) $\left[\frac{mm}{s^3}\right]$	27.74	18.97	27.78	22.25	8.51	4.97	9.02	4.87
Arc Length $[mm]$	31.74	16.75	34.82	20.61	22.87	9.20	24.17	11.92
Time $[s]$	34.42	26.32	36.68	34.92	6.78	4.63	7.64	5.01
Effort $\left[N'\right]$	75.33	69.59	88.88	100.00	33.64	20.48	36.59	24.70
Speed $\left[\frac{mm}{s}\right]$	0.87	0.58	1.15	0.92	3.23	2.25	2.96	1.94

2.3.1. Performance Criteria

Jerk A trajectory with a maximum smoothness will result in a maximal movement efficiency [14] and ensures a reduced interaction effort from the user's side. Hence, the Human-Robot interface gets improved [64]. It can be quantified as a function of jerk given by Eq. 2.2, which is the time derivative of the acceleration. The jerk is calculated for the force and also for the pose profile. The smoothness of both is important for the interaction.

$$\dddot{X}_i = \left(\frac{d^3 X_i}{dt^3}\right) \ , j = \sum_{i=1}^{i=n} \left\|\dddot{X}_i\right\|_2 \ , X_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \tag{2.2}$$

Analysis of the jerk is based on the force profiles. Regarding *Top* the mean jerks $\left[\frac{N}{s^3}\right]$ for *Adaptive KG* $M_{KA} = 960$ and for *KG* $M_{KG} = 1330$ are much smaller than for *Keyboard* $M_K = 12847$ and for *Space Mouse* $M_S = 14197$. Similarly, jerk analysis of the pose profiles w.r.t. *Top* shows that the mean jerks $\left[\frac{mm}{s^3}\right]$ $M_{KA} = 6.39$ and $M_{KG} = 7.99$ are much smaller than the mean jerks for the other two modes $M_K = 45.61$ and $M_S = 52.85$. For *Bottom*, the jerks of the force profiles are $M_{KA} = 1725$, $M_{KG} = 1506$, $M_K = 7007$, $M_S = 7718$ and for the pose profiles the jerks are $M_{KA} = 9.02$, $M_{KG} = 8.51$, $M_K = 27.74$, $M_S = 27.78$ respectively.

Arc Length This measure gives information about the total length traversed while moving along the given trajectory. High arc length values can be interpreted as a deviation from the intended or optimal path. It is assumed that participants also try to use the optimal path. Here, the arc length is calculated by Eq. 2.3.

$$S = \sum_{i=1}^{i=n} \sqrt{\Delta x_i^2 + \Delta y_i^2 + \Delta z_i^2} \tag{2.3}$$

The mean arc lengths $[mm]$ of the sampled trajectory for *Top* are $M_{KA} = 37.34$, $M_{KG} = 40.72$, $M_K = 53.62$, $M_S = 65.00$. For *Bottom*, the mean arc lengths are $M_{KA} = 24.17$, $M_{KG} = 22.87$, $M_K = 31.74$, $M_S = 34.82$.

Time of Completion Time of completion is a good benchmark criterion for comparing the performance of a system under different constraints. The mean times $[s]$ of completion

for *Top* in the considered modes are $M_{KA} = 4.14$, $M_{KG} = 5.96$, $M_K = 60.38$, $M_S = 67.62$. At *Bottom*, they are $M_{KA} = 7.64$, $M_{KG} = 6.78$, $M_K = 34.42$, $M_S = 36.68$.

Total Effort The total effort $[N]$ needed to recover from the errors corresponds to the amount of energy consumed by the system. This benchmark criterion gives an insight on the efficiency of task execution under different interaction modes. The sum of forces over an entire task is taken and scaled down to a scale of 0 to 100 for easier comparison. This scaling has been done w.r.t. all task executions of a specific assembly error and an applied interaction method.

$$E = \sum_{i=1}^{i=n} \|F_i\|_2 \,, \; F = \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} \tag{2.4}$$

The mean efforts spent on *Top* in each mode are $M_{KA} = 7.30$, $M_{KG} = 15.23$, $M_K = 54.99$, $M_S = 100$. Recovering from *Bottom* they are $M_{KA} = 36.59$, $M_{KG} = 33.64$, $M_K = 75.33$, $M_S = 100$ respectively.

Average Speed The average speed $\left[\frac{mm}{s}\right]$ gives an idea about how fast a workpiece is moved during the task execution. Higher speed means the task is being solved in less time.

The mean Cartesian speeds during recovery from *Top* for each interaction mode are $M_{KA} = 8.80$, $M_{KG} = 8.46$, $M_K = 1.04$, $M_S = 1.27$ respectively. During recovery from *Bottom* they are $M_{KA} = 2.96$, $M_{KG} = 3.23$, $M_K = 0.87$, $M_S = 1.15$ respectively.

User Satisfaction User satisfaction is a qualitative analysis method that captures affective perceptions of participants while using the system in question. It represents the degree of favourableness the participant shows w.r.t. the system [85]. Participants' satisfaction is measured by ratings. Each participant completed a questionnaire after the experiment to determine their satisfaction. Here, the rating was in a scale from 1 to 4 where 1 means good and 4 bad. In the questionnaires, no distinction was made between both assembly errors. So this criterion refers to the application of a method as a whole.

The results of the participant satisfaction rating for the both assembly errors are as following, $M_{KA} = 1.48$ and $SD_{KA} = 0.88$, $M_{KG} = 1.77$ and $SD_{KG} = 0.95$, $M_K = 3.23$ and $SD_K = 0.84$, $M_S = 2.97$ and $SD_S = 1.96$.

Measure of Intuitiveness Here, learning effects can be defined as the process in which the performance of a participant improves by performing the experiment repeatedly. Intuitiveness can be expressed as a function of a learning effect. So it can be argued that in an intuitive scenario, the performance of a participant would already be much better [49]. A learning curve gives a glimpse into the intuitiveness of the control modes and Fig. 2.5 shows the learning curves of all considered interaction modes while recovering from *Top*. A larger variation in a learning curve indicates that it is difficult for a participant to get used to a mode and the interaction is counter-intuitive. On the contrary, a constant learning curve is a sign of an intuitive system where the participant's intuition will lead them automatically to the best solution.

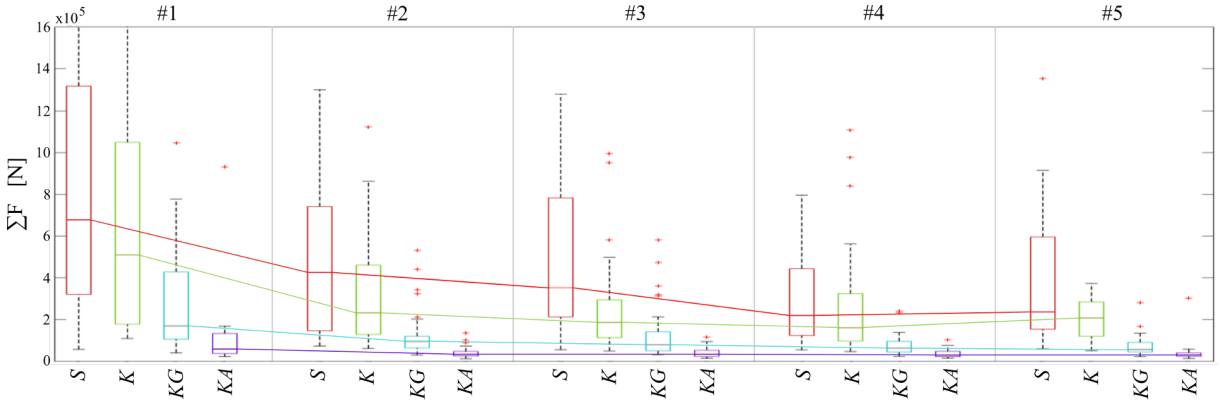


Figure 2.5.: Learning curves for *Top*. These curves are based on the summed force of all participants for each trial (#). The course of each curve shows the learning effect. In addition, there are differences between the input devices (***S****pace Mouse*, ***K****eyboard*, ***K****inesthetic* ***G****uidance*, ***K****inesthetic Guidance* ***A****daptive*) regarding the learning effects.

2.3.2. Discussion

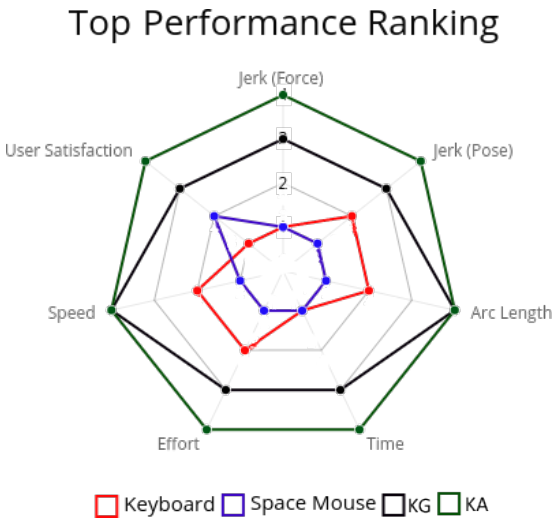
It is observable from the numerical results in Sec. 2.3 that the motions w.r.t. both *KG* modes are smoother than the other interfaces like *Keyboard* or *Space Mouse*. While using *Keyboard* or *Space Mouse*, the jerks in force profiles are comparable and both are much higher than jerks produced while using the *KG* methods. A comparison of the two *KG* methods shows that regarding *Top*, *Adaptive KG* is superior to normal *KG*. Hence, considering *Bottom* both are comparable. The jerk analysis of the sampled pose trajectories also yields to similar results where the *KG* methods are far better than their counterparts, here again the *Adaptive KG* has the lowest jerk regarding *Top* and both *KG* methods have comparable results considering *Bottom*.

The arc length of both *KG* methods are shorter compared to *Keyboard* or *Space Mouse*. This means that the participants are able to recover from errors much faster and better while using the *KG* methods. A brief look into the *KG* methods shows that *Adaptive KG* has smaller arc length in both *Top* and *Bottom*. Similarly, the time of completion for both tasks shows a huge difference between the considered interaction methods. The *KG* methods are several times faster than the other methods. Regarding *Top* the *Adaptive KG* is better and considering *Bottom* both *KG* methods have similar performance. There is no distinct advantage between *Space Mouse* or *Keyboard* since the time of completion is comparable in both cases. The results also show that the effort in task completion needed by the *KG* methods is much smaller than the *Space Mouse* and *Keyboard*.

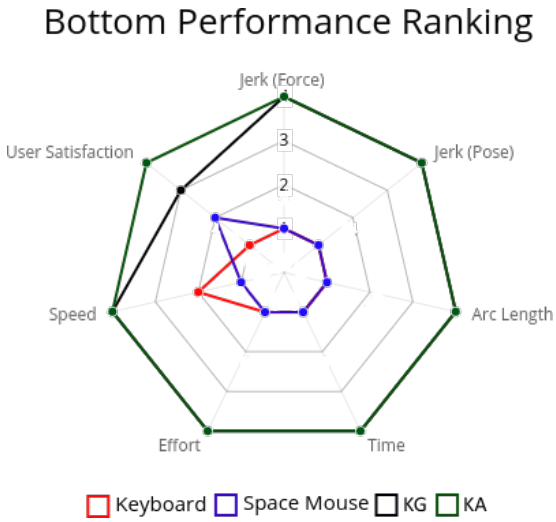
The Cartesian speed of both *KG* methods is nearly the same. Only *Keyboard* and *Space Mouse* are much slower. This can be attributed to the human knowledge of how to solve a task intuitively while interacting kinesthetically. The Cartesian speed with *Space Mouse* is slightly better than using *Keyboard*.

The intuitiveness and participant satisfaction are subjective measures which give information about interaction quality. Here the *KG* methods are clearly superior to the *Keyboard*. Participants' satisfaction with *Keyboard* seems to be the worst and with the *Adaptive KG* is the best. The learning curves from Fig. 2.5 suggest that both *KG* modes are intuitive compared to the other input methods. There is a high variation in the learning curves on *Space Mouse* and *Keyboard* while the variation w.r.t. *KG* methods is minimal.

The performance of all interaction modes is ranked w.r.t. their statistical significance based on Kruskal-Wallis ANOVA in Fig. 2.6(b) and in Fig. 2.6(a). The graphs show that the *KG* methods always have a clear advantage over the others. For both errors, the performance of *Keyboard* and *Space Mouse* remains similar. The *Keyboard* has a slightly better performance as *Space Mouse*. For *Top*, clearly *Adaptive KG* is much better, but regarding *Bottom* adaptation of stiffness might not be necessary since there is no statistical difference in the performances except in case of participant satisfaction. One possible reason might be that *Bottom* is easier to solve (cf. Sec. 2.1.2). Therefore, it can be assumed that no stiffness adaption is needed in this case.



(a) This figure shows the ranking of different performance criteria for *Top*. It is based on the statistical significance calculated using the Kruskal-Wallis test.



(b) This figure shows the ranking of different performance criteria for *Bottom*. It is based on the statistical significance calculated using the Kruskal-Wallis test.

Figure 2.6.: Performance rankings as spider plots.

2.4. Recommendation of an Input Device

Generally speaking, results (cf. Sec. 2.3) show that *KG* methods are the best for interaction with a robot in terms of both performance and participants' satisfaction regarding an industrial assembly scenario. This could be attributed to the fact that participants get direct tactile feedback during the interaction. So, it makes the *KG* methods more intuitive and easier. That claim can be verified by the results. Both *KG* methods show good performance regarding different criteria, e.g. they are at least 10 times better w.r.t. the time of completion. In addition to that, the analysis of learning curves shows that these methods are the most intuitive way for interacting with the robots. Even a small amount of trials results in satisfactory interaction and superior performance. Hence, these methods will be suitable for untrained and naive participants. However, using stiffness adaptation offers added value and justifies the effort depends on the respective task.

Chapter 3

Characteristics of Kinesthetic Guidance during Assembly

In Chap. 2 it was shown that *KG* and also *KG* with stiffness adaptation are well suited to be used for *HRI* in industrial assembly scenarios. Unfortunately, it is the case that *KG* with stiffness adaptation is not possible by default with standard robot controllers. In the best case, only *KG* without any adaptation is supported by such a controller. Therefore, the focus is on *KG* in a non-adaptive form. This chapter is based on the previous publication [58].

Although the presented error handling approach (cf. Sc.1.3) utilises *HRI* for demonstrating a recovery from an assembly error to a robot. Nevertheless, complete assembly operations are considered here. The reason for this is that individual errors cannot be exactly reproduced if the assembly is carried out by hand. Thus, a meaningful identification of *KG*'s characteristics is only possible by analysing complete assembly operations and comparing them with manual assembly.

The question that arises through the suitability of *KG* in *HRI* (cf. Chap. 2) is whether a human can perform the same tasks and in an equal quality by guiding a robot as performing the task without a robot.

3.1. Hypothesis

Since the focus is on the applicability of *KG* in assembly scenarios, the characteristics of *KG* need to be known. Therefore, a user study was carried out and the following four hypotheses are considered:

1. **Learning effect by repeated execution:** Executing a task or several tasks that resemble each other repeatedly reduces the required time significantly.

2. **Manual assembly performance cannot be achieved by *KG*:** The duration for completing a task required in manual assembly cannot be achieved when guiding the robot.

3. **Performance in an assembly task depends on personal attributes:** The required time varies with age, spatial sense, previous knowledge and personal attitude towards technical devices.

4. **Correlation between performance in manual assembly and *KG*:** Participants performing well in manual assembly will also perform well in the *KG* trials.

3.2. Experimental Setup

Within this section a description of the experimental setup including the used parts, sensors and the robot is given. This is followed by a description of the study design including experimental procedure, questionnaire and composition of the participants. Finally, a study specific performance measure is introduced.

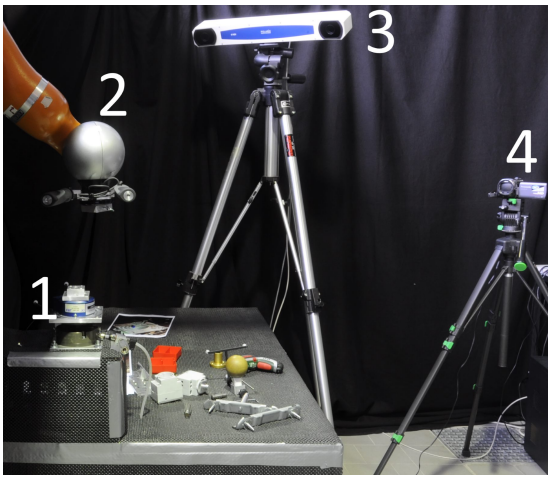
3.2.1. Hardware

An overview of the experimental setup is given by Fig. 3.1(a). In addition, a detailed view of the force-torque sensor and the basis platform for carrying out the experiments is given by Fig. 3.1(b). The robot (2), specifically a KUKA Light Weight Robot (LWR IV+) [11], was mounted upside down to minimize interference between participants and the robot. All assembly tasks were executed on a basis platform (5). This one was mounted on top of a JR3 50M31A-I25 force-torque sensor (1). An overload protection device (6) is visible under the force-torque sensor. This device was used to avoid potential damage to the force-torque sensor due to overload.

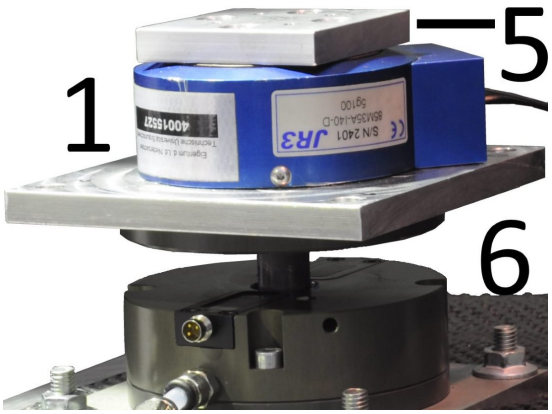
It was possible to measure acting forces and torques for all assembly methods by using the additional force-torque sensor. Otherwise, it would have been only possible to measure forces or torques if the robot was used for an assembly. During the usage of the robot, the end-effector pose was also acquired directly via the KUKA Fast Research Interface. An optical tracking system (3), the Polaris Accedo by Northern Digital Inc., was used to track the pose while executing tasks in combination with the handle (cf. Sec. 3.2.3). During the

entire experiment, the participants were recorded on video with a camera (4). The whole setup was surrounded by a black curtain, as shown in Fig. 3.1(a), to generate reproducible environment settings with minimal disturbance from outside for each participant (cf. Fig. 3.2).

Due to the fact that the tracking system works with infrared light, most of the reflecting surfaces, e.g. the metallic table out of Fig. 3.1(a), needed to be covered with non-reflecting rubber. Otherwise, the tracking system could not be used reliably.



(a) Overview of the complete experimental setup.



(b) Close up of the force-torque sensor and the basis platform.

Figure 3.1.: Experimental setup of the user study.

Figure 3.2.: Snapshot from the user study: Showing that the setup is surrounded by a black curtain to minimize external disturbance and to create a reproducible environment.

3.2.2. Assembly Tasks

This user study is based on four assembly tasks:

- *Peg:* A peg which has to be inserted into a hole.
- *Spline Shaft:* A spline shaft which has to be inserted into a corresponding base.
- *DIN Rail:* A socket with a clip needs to be mounted onto a *DIN* rail.
- *Bracket:* A mounting bracket has to be assembled in such a way that it fits to an associated base.

Exemplary images of all assembly tasks are shown in Fig. 3.3. Detailed engineering drawings of all used parts can be found in Appendix A.1, A.2, A.3 and A.4.

(a) *Peg* (b) *Spline Shaft* (c) *DIN Rail* (d) *Bracket*

Figure 3.3.: Exemplary images of all four assembly tasks. The transparent version shows the workpiece before assembly. The coloured version shows the result.

3.2.3. Assembly Methods

Three different methods (*Robot*, *Hand* and *Handle*), which are depicted in Fig. 3.4(a), Fig. 3.4(b) and Fig. 3.4(c), are used to perform the assembly tasks. For the method *Robot*, the assembly parts are attached to the robot's end-effector. The robot was controlled in gravity compensation mode. This mode does not compensate inertia and friction completely. The robot was equipped with two handles at the end-effector enabling full control of orientation and position by a human operator. However, the hands have been used directly for the method *Hand*. This is also referred to as manual assembly. For the method *Handle*, the assembly parts were attached to a handle. This method was supposed to be an intermediate method between *Hand* and *Robot*. Hence, *Handle* allows limited tactile feedback in the assembly but less interference from inertia or damping than a robot. Neglecting inertia and friction, the methods *Handle* and *Robot* differ merely with respect to their interface (two handed vs. one handed).

(a) *Robot* (b) *Hand* (c) *Handle*

Figure 3.4.: Illustration of all assembly methods used in this user study.

3.2.4. Order of Experiments

Different combinations and orders of assembly tasks and methods were performed for detecting learning effects. Possible permutations had to be limited in order to avoid an exponential blow-up. All four tasks (*Peg*, *Spline Shaft*, *DIN Rail*, *Bracket*) were performed in the mentioned and also in reversed order. For each assembly method, two orders (*Hand*, *Robot*, *Handle*) and (*Robot*, *Hand*, *Handle*) were investigated. Only one of these combinations has been performed by a participant. For each task, the participants cycled through the different methods before switching the assembly task. A special term is used for a specific combination of a method and a task like e.g. *Robot/Peg*. During the study, each participant performed all twelve combinations. Each combination was used five times in a row before the next combination has been chosen.

3.2.5. Participants

The total number of participants is 78. Two thirds of them are between 14 and 39 years old. The remaining third is between 44 and 71. 38 are female, 40 male. The exact distribution in terms of age and gender is shown in Fig. 3.5. Nearly half of the participants are university graduates or students. Thus, it cannot be assumed that these participants describe a representative sample of the population.

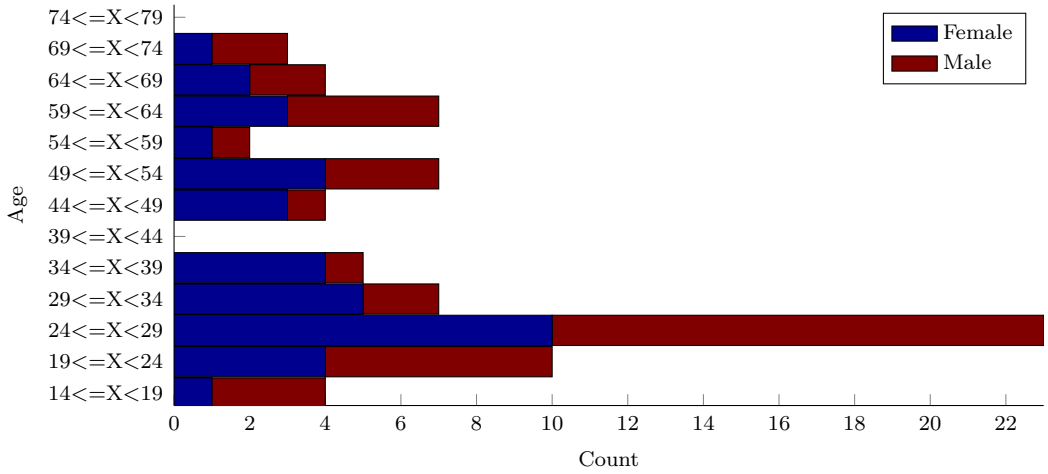


Figure 3.5.: Distribution of the participants regarding their age and gender.

3.2.6. Definition of a Contact Phase

A *contact phase* is defined as that part of an assembly operation where contact between the used workpieces is established. It is possible that an assembly operation contains multiple contact phases for example if it was not initially possible to insert a peg into a hole. Here, the number of contact phases and also their temporal length are used as a performance measure in the evaluation of the experimental data. Therefore, an automatic contact phase detection process is needed. Between the initial contact and the loss of contact the signal-to-noise ratio of the measured forces is suitable be used as input of the detection process. If these forces are above the sensor noise level for at least 0.1 seconds it is assumed that the workpieces are in contact. Otherwise, it is assumed that there is no contact if the measured forces are in the range of the sensor noise for at least 1.0 seconds. These threshold values are based on observations using the video recordings and a frequency analysis of participant motion (cf. Sec. 3.3). Using this definition of a *contact phase*, up to 97% of the experiments had only one contact phase. This means, after establishing initial contact, the participants rarely lost contact deliberately to retry the whole process.

3.3. Experimental Results

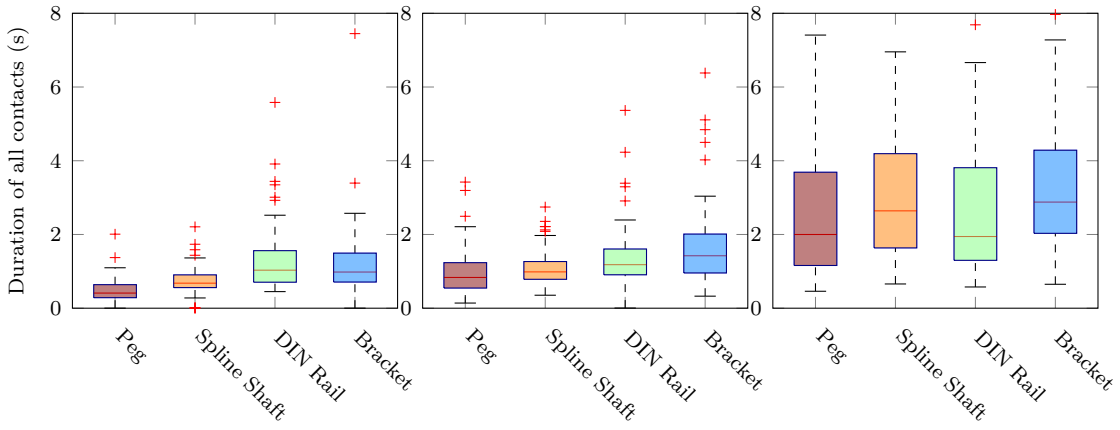
In the following presentation of the results, box plots are used. If the data points followed a Gaussian distribution, the dotted lines would cover 99.3% of the samples. However, the

experimental data is not always distributed normally. If not mentioned otherwise, the number of data points in each box corresponds to 78 – the number of participants. Plots are scaled that the dotted lines are always visible, but not necessarily every outlier.

3.3.1. Complexity of Assembly Tasks

For all combinations of assembly tasks and methods the contact duration is evaluated in order to give a rating of the task complexity. When looking directly at the contact duration, only those of the last trials should be considered. The contact durations of trial 1 to 4 can be corrupted through learning effects (cf. Sec. 3.3.2). In Fig. 3.6, the contact durations of the last trial are visualized. It can be seen in Fig. 3.6(a) and Fig. 3.6(b) that the assembly tasks have roughly an increasing duration from task *Peg* to *Bracket*. This progression suggests that the difficulty of the tasks also increases in that order.

When using *Robot*, the contact duration is longer and more scattered compared to using *Hand* or *Handle*. This observation suggests that the parameters of the assembly strategy using *Robot* differ from the natural ones. There is also a significant difference in the order of complexity in the assembly tasks *Spline Shaft* and *DIN Rail*. This might be due to the increased lever when using the robot.

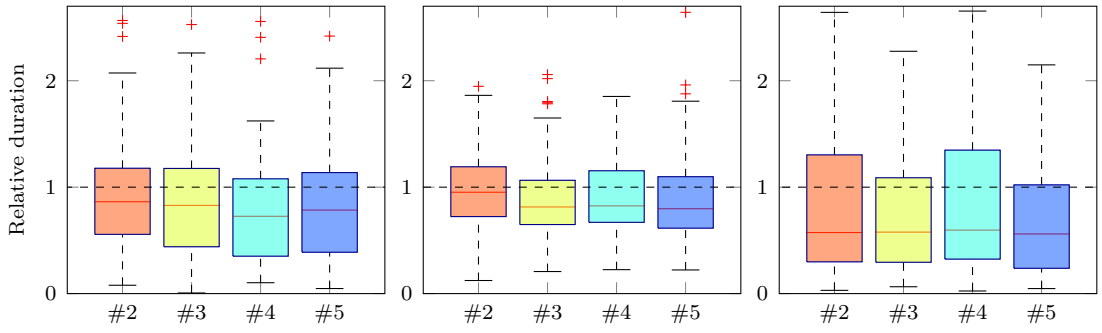


(a) *Hand* (b) *Handle* (c) *Robot*

Figure 3.6.: Duration of contact in the fifth trial for each combination of assembly method and task.

3.3.2. Learning Effect

Regarding the learning effect by repeating the same experiment, there is a distinctive decrease in the *contact phase* duration from the first to the second trial. All following trials only improve the duration by a small amount. Fig. 3.7 shows the relative duration of the second to the fifth trial of the task *Peg*. For each participant, the durations of the last four trials are divided by the duration of the first trial yielding a contact duration normalized to the first trial. The intention of this approach is to make the learning effects of tasks comparable. For example, a participant who requires 6 seconds in the first trial and only 3 seconds in the second trial would score 0.5 in the second trial. Using this figure, the individual relative changes can be identified. The largest relative improvement occurs in the method *Robot* where the median improves by about 43%. The improvement from the first to the second execution in the method *Hand* is only about 23%. These observations confirm the ease of the learning attributed to *KG*.



(a) *Hand* (b) *Handle* (c) *Robot*

Figure 3.7.: Relative duration (to first trial) of contact phases for assembly task *Peg*.

Note that due to the relative visualization, the variance in the first trial is set to zero but is still included because the relative durations of the following trials show higher variances accordingly. The other assembly tasks exhibit a similar development and are omitted due to space restrictions.

Fig. 3.8 shows the results for the combination *Robot/Peg*. Here, only that half of the participants who started with the *Peg* are considered. This subset is further divided into two groups with about 20 persons: A group who used the robot first (cf. Fig. 3.8(c) & (d)) and a group who performed manual assembly first (cf. Fig. 3.8(a) & (b)). Consequently, this combination was the first one for participants in (c) & (d) and the second one for participants in (a) & (b) (following their combination *Hand/Peg*).

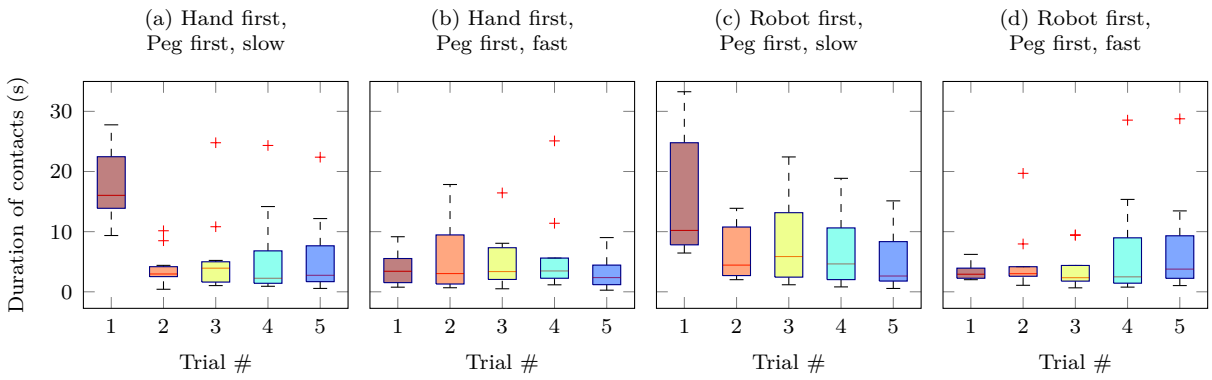


Figure 3.8.: *Robot/Peg*: Comparison of the development of 'fast' and 'slow' participants and influence of method order.

Both of these groups are further divided into the 50% fastest (cf. Fig. 3.8(b) & (d)) and the remaining (cf. Fig. 3.8(a) & (c)) group, according to their result in the first trial of *Robot/Peg*. Therefore, each box in the plots consists $10 - 12$ data points, representing participants.

An interesting fact is that the median duration in the last trial of all four groups is similar (2.7 s, 2.4 s, 2.6 s, 3.8 s) compared to the standard deviation. This means that after five trials of *Robot/Peg* members of each group can achieve a similar result (regarding duration) even if they did not have previous knowledge of the task (group that used the robot first) or had shown bad performance in the first trial.

3.3.3. Statistical Tests

It was expected that the experimental results would depend on the one hand on individual attributes like technical affinity and spatial sense, and on the other hand on demographic data like age. Information regarding individual attributes of the participants was acquired by using a questionnaire. It contains standardised questions (cf. [49]) to measure the self-assessed technical affinity and images of $3D$-tasks for the spatial sense. Statistical tests are used to test correlations between all recorded attributes.

A standard ANOVA cannot be applied as the random variables are not distributed normally. However, the Kruskal–Wallis one-way analysis of variance - a non-parametric version of ANOVA is used. This statistical test does not rely on the normal distribution assumption [43]. The test checks the null hypothesis that two or more sets of samples are

from the same distribution. The alternative hypothesis is that not all samples are from the same distribution.

Here, the contact duration, the average mating force, and the length of the assembly path as samples are used. The partition into groups is based on various factors including technical affinity, spatial sense, age, education, and other demographic parameters. The tests are conducted on data of individual combinations, e.g. *Robot/Peg*. The reason is that it has already been shown that the results are different for each combination. The statistical test results show no general conclusive dependency on any of the tested group partitions. Several tests reject the null hypothesis at a 5% significance level but these rejections only occur in single experiments. Therefore, no significant general influential factors can be found.

As an example, the influence of spatial sense on the contact duration is visualized in Fig. 3.9. Here, the cumulative relative frequency that a member of a group has successfully finished the task after the given time is depicted. The three groups are created based on the individual spatial sense score and are approximately equal in size. In the left case of that figure (*Robot/Peg*), the Kruskal–Wallis test does not result in a significant rejection of the null hypothesis, in the right case it is rejected at the 5% significance level, meaning that the samples are not drawn from the same distribution.

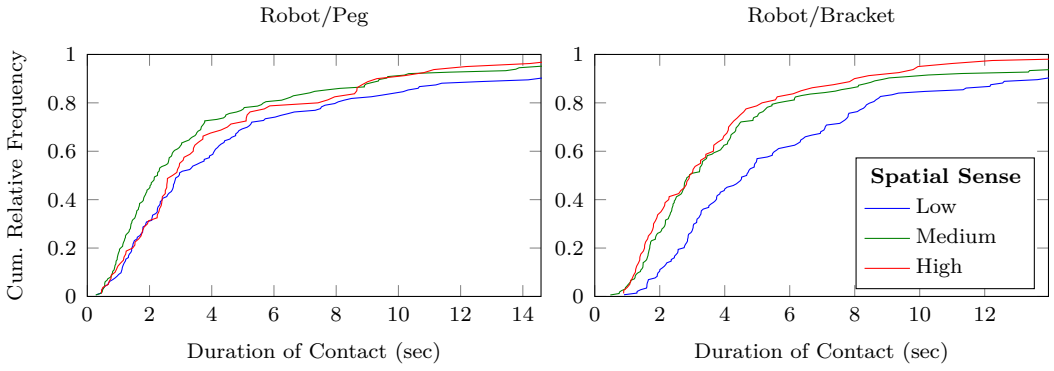


Figure 3.9.: Probability of being below a certain contact duration based on spatial sense classifications of the participants

3.3.4. Correlation between the Assembly Methods *Hand* and *Robot*

The hypothesis that participants showing good results by using *Hand* also achieve good results by using *Robot* is verified here. Therefore, the participants are divided into two groups. For each participant and for each assembly task, the median duration of the five *Hand* trials is calculated. Then the median of these medians is taken as the separator

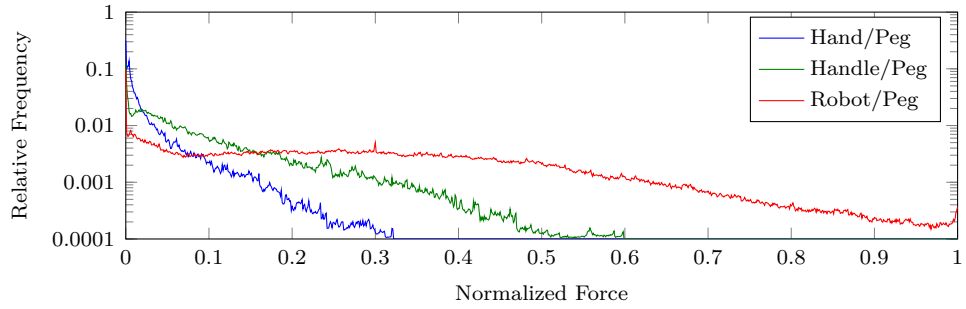
between the two groups. Note that these two groups contain no information about the order of assembly tasks or methods.

Looking at the two groups and comparing the median contact duration of the *Robot* trials, only in the task *Bracket* is a significant difference between the groups can be observed. This fact means that people who complete *Hand* trials faster are not in general faster in completing *Robot* trials. The results of the Kruskal-Wallis tests are $p = 0.5$ (*Peg*), $p = 0.2$ (*Spline Shaft*), $p = 0.6$ (*DIN Rail*) and $p = 0.05$ (*Bracket*). Apart from the *Bracket*, all cases suggest that the division of the groups is not significant at a 5% level, meaning that there is no conclusive evidence for a correlation between the contact duration in *Hand* and *Robot* trials.

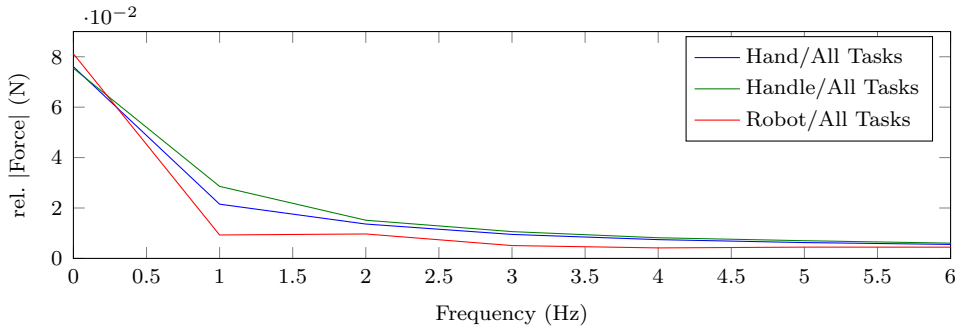
3.3.5. Differences between Assembly Methods

In order to compare different assembly methods, force-torque data is used since this type of data was recorded for all assembly methods. Comparing the applied forces reveals significantly different distributions. In Fig. 3.10(a) the relative frequencies of forces for the task *Peg* executed with each method are shown. Each line represents a sum of histograms from all relevant trials in which the forces were normalized to the maximum force applied by each participant (over all three methods). Additionally, the bin counts of these histograms were normalized with respect to the duration of the corresponding trial to ensure equal influence of each trial.

The first major difference is the measured maximum force. Using *Robot*, higher mating forces were applied to the parts compared to the methods *Hand* and *Handle*. This might be due to physical effects like the higher inertia and friction loss in the robot joints. However, the method *Handle* shows a similar, but less pronounced, behaviour which indicates that the decreased tactile feedback has a dominant influence. Other possible factors are the different points of force application and the deteriorated vision due to the robot. The different shapes of the distributions are another striking difference. For the methods *Hand* and *Handle* the distributions decrease monotonically whereas the method *Robot* has a long phase of nearly constant relative frequency after an initial drop.



(a) Relative frequencies of forces for different methods executing task *Peg*



(b) Force amplitude spectrum during participants motion

Figure 3.10.: Comparison of different assembly methods w.r.t. the occurring frequencies.

For each assembly method, the force data of all participants and all experiments is used to perform a Fast Fourier Transform (FFT). Fig. 3.10(b) shows the mean of all FFTs per assembly method. Before the mean is calculated, the amplitude of each FFT is normalized by using the integral of the corresponding FFT. Without this normalization, participants who mainly use high forces would have a greater influence on the final shape. The steady force component of the method *Robot* is the highest and the corresponding plot has the steepest downward slope. This might be due to the high inertia of the robot and the friction losses in the robot's joints resulting in a mechanical low-pass. In contrast to the method *Robot* the shapes of the methods *Hand* and *Handle* are similar in shape. One interesting detail is that the shape of the method *Handle* is a bit higher than the shape of the method *Hand*, except for very low frequencies.

3.4. Review of the Characteristics

Revisiting the hypotheses from Sec. 3.1, important findings of the user study are summarized. Substantial evidence for rapid learning effects could be found confirming the ease

of learning attributed to *KG*. Surprisingly, neither a general correlation between the performance in manual assembly and in *KG* nor a clear dependency on personal attributes could be observed.

A learning effect exists, especially regarding the first and the second execution. The effect is larger in experiments involving a robot. More complex assembly tasks could require more repetitions before a significant learning effect is observable. Such an effect also occurs when executing the same assembly task by different assembly methods. Participants who assemble manually before using *KG* perform better in their first robot trial concerning the same task. However, this performance gain decreases gradually and cannot be observed in the final trial. After several repetitions of a specific task, participants achieve similar performance. This observation indicates that *KG* is a suitable method for *HRI* in assembly scenarios.

Solving an assembly task manually results overwhelmingly often in the best performance regarding contact duration and applied forces. The relative complexity difference between tasks observed in manual assembly cannot generally be transferred to *KG*. There was no conclusive evidence that the performance in an assembly task depends on personal attributes in general. Merely a slight dependence on spatial sense could be found.

Participants who show good performance in manual assembly do not necessarily perform well when using *KG*. No general statistical evidence for a correlation between the performance of *KG* and that one of manual assembly could be observed. Furthermore, the user study suggests that human assembly strategies cannot be extracted easily via *KG*. Comparing manual assembly with *KG*, substantial differences w.r.t. contact duration, applied forces, and movement frequencies can be observed. Thus, deriving human assembly strategies by *KG* is likely to yield significantly distorted results. Extracting and transferring strategies to robots using *KG* is possible. But, the effects caused by reduced tactile information, inertia and friction losses of the robot have to be considered.

Chapter 4

Assembly Task Representation

A formal and abstract representation of assembly operations offers the possibility to monitor an assembly. Therefore, errors can be detected or even predicted. Also, error classification is possible so that error specific recovery strategies can be applied. The Hierarchical Decomposition (*HD*) has been proposed in [59] as such a representation which can be used for assembly monitoring. In [61, 62] the *HD* gets extended so that an application for error classification and also for execution of recovery strategies is possible. This chapter is based on the already mentioned previous publications.

The main application purpose of the *HD* is a formal and abstract representation of an assembly operation w.r.t. task specific knowledge and experience of human domain experts. In order to meet this purpose, the *HD* is designed as an rooted and non-balanced tree which is called decomposition tree (cf. Sec. 4.1.1). The leafs of this tree represent different states of the assembly operation. Conditions for state transition guarantee that the state of an assembly is known. States and conditions are specified by a domain expert taking into account his task specific knowledge. The decomposition tree can reach different levels. So, it is possible to depict parts of the operation in detail, which are especially interesting or problematic for a domain expert without increasing the overall complexity of the decomposition. Thus, the *HD* represents a process model for describing an assembly operation and also for online error detection and prediction during assembly. Therefore, error states have to be included in the *HD*.

Additional information e.g. a mating axis must be defined for each error state to increase the *HD*'s applicability. Due to the additional information it is possible to select and to apply a suitable recovery strategy (cf. Chap. 5) and also to recover from errors during automated assembly. Here, it is important to distinguish that a domain expert's task specific knowledge is used for defining the *HD* of an assembly operation. However, shop-floor workers' task specific knowledge and experience is not encode in the *HD*. Instead,

it is encoded in the taught recovery strategies (cf. Sec. 1.3) which are reused for similar errors.

4.1. Definition of the Hierarchical Decomposition

First a description of the decomposition tree and its properties is given in Sec. 4.1.1. Subsequently, the behaviour of the states during operation is described in Sec. 4.1.2. Since the *HD* should not only be used for process monitoring, but also for recovering from errors, additional state attributes (cf. Sec. 4.1.3) are needed. These form the basis for selecting and executing situation specific recovery strategies (cf. Chap. 5).

4.1.1. Decomposition Tree

The *HD* of an assembly operation is depicted as a rooted and non-balanced decomposition tree in Fig. 4.1. Each leaf level of that tree represents a hierarchical level of the *HD*. The root is located at the first hierarchical level L_1 of the decomposition. It represents a whole assembly operation. In contrast, only parts of an operation are represented by the leaves distributed to different hierarchical levels $L_k, k \in \mathbb{N}$. In this monograph, the index $_k$ is always used to show to which level an element belongs to. Each leaf depicts a state $\mathbf{S}_{j,k}, j \in \mathbb{N}$, of the considered assembly operation. Expect for L_1, there can be $n_k, n_k \in \mathbb{N}$, states per level L_k. Each state $\mathbf{S}_{j,k}$ has $m_{j,k}, m_{j,k} \in \mathbb{N}_0$, substates $\mathbf{S}_{j,k+1}$ at level L_{k+1}. Apart from the root, each state $\mathbf{S}_{j,k}$, which is connected to at least one substate, has also a special substate $\mathbf{S'}_{j,k+1}$. A property of $\mathbf{S'}$ is that this state has no substates. It is always selected, if none of the other substates $\mathbf{S}_{j,k+1}, i \neq j$, of $\mathbf{S}_{j,k}$ can be selected. In order to obtain an accurate description of an assembly operation, the lowest possible leaves of the decomposition tree are used for describing the operation. But it cannot be assumed that a *HD* contains all conceivable states. This is why a special state $\mathbf{S'}_{i,k}$ is selected whenever a suitable state is missing. If states are defined for a level L_{k+1} and there is a missing state definition, a return to lower hierarchical levels can be prevented by choosing $\mathbf{S'}_{i,k+1}$. Special states are also used for supporting the domain expert while decomposing an assembly operation (cf. Sec. 4.3).

Beside of the hierarchical state structure, the *HD* contains also conditions for specifying state transitions. The current state of an assembly operation is always clearly defined by evaluating these conditions. Such a condition $C_{i,j,k}$ specifies a state transition from $\mathbf{S}_{i,k}, i \in \mathbb{N}$, at level L_k to $\mathbf{S}_{j,k+1}$ at level L_{k+1}. The indices i and j of a condition describe which states are involved and the initial level is given by the index $_k$.

The condition $C_{i,j,k}$ is defined as follows:

$$C_{i,j,k} = C_U \wedge C_R$$

C_U : User defined condition for state transition between two states $\mathbf{S}_{i,k}$ and $\mathbf{S}_{j,k+1}$ on different hierarchical levels L_k and L_{k+1}.

C_R : User defined precondition for ensuring that a state transition is only possible after reaching a defined step in the assembly operation. If C_R is not explicitly defined, it is always *true*.

All conditions are evaluated in parallel to an assembly operation. C_U and C_R return either *true* or *false*.

$A_k = \{C_{i,j,k} | i \in \{1, ..., n_k\}, j \in \{1, ..., n_{k+1}\}$ is a set of conditions $C_{i,j,k}$ at the same hierarchical level L_k. An important point is that only one condition $C_{i,j,k} \in A_k$ can be *true* at the same time. Otherwise, it would not be possible to unequivocally represent the current state of an assembly operation at a certain time. If a condition $C_{i,j,k}$ is not *true* any more (due to the proceeding of an assembly operation), there is a return to state $\mathbf{S}_{i,k}$ and all conditions $C_{i,j,k}$,$j \in \{1, ..., n_{k+1}\}$, are evaluated to determine the current substate.

An exemplary decomposition tree containing different hierarchical levels, multiple states per level and conditions for state transition is given by Fig. 4.1.

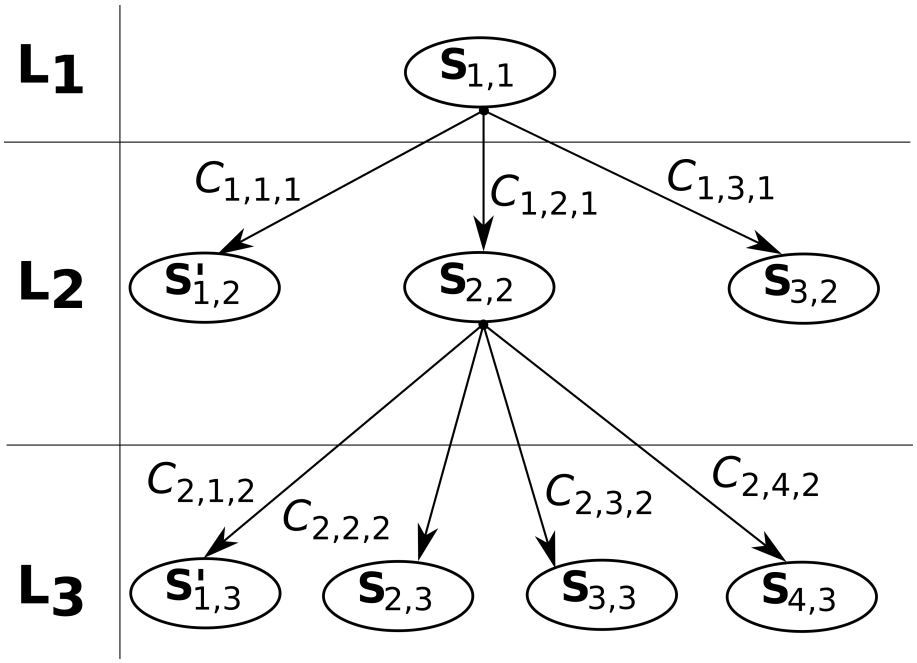


Figure 4.1.: Hierarchical decomposition *HD* of an assembly operation. States $\mathbf{S}$ are distributed on three different hierarchically arrange level $L_i, i \in \{1, 2, 3\}$. Conditions C define state transitions so that there is one active or selected state. The *HD* depth depends on the characteristics of the assembly process.

4.1.2. Runtime Behaviour

A representation overtime is more intuitive for showing the dynamic behaviour of states during an assembly operation. If error states are defined within the *HD* it can be used for monitoring an assembly and especially for error detection. Such a representation for an exemplary schedule of an operation as described by the decomposition tree in Fig. 4.1 is shown in Fig. 4.2. It should be emphasized that there is a break on L_3 because the special state $\mathbf{S'}_{1,2}$ has no substates and the domain expert has not defined substates of $\mathbf{S}_{3,2}$. This shows how the decomposition tree adapts to the interests of the domain expert.

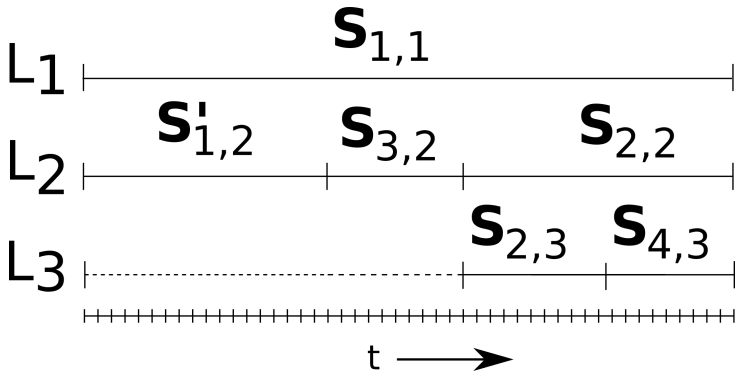


Figure 4.2.: Exemplary schedule of an assembly operation. The basis of this example is the decomposition tree shown in Fig. 4.1.

However, the representation over time also offers the possibility to observe which condition is *true* at which time. The conditions which are *true* at time t_s are part of a set A and the conditions which are *true* at time t_{s+1} are part of a set B. By assuming that the state switches from $\mathbf{S}_{j,k}$ to $\mathbf{S}_{i,k}$ $(i \neq j)$ between time t_s and t_{s+1}, the sets A and B will contain different conditions. These $\{C_T | C_T \in B \wedge C_T \notin A\}$ describe what has changed and what has triggered the state transition.

4.1.3. State Attributes

During an assembly operation, the *HD* can be used to monitor assembly progress and to detect or classify types of errors according to predefined states. However, for being able to use the *HD* for error handling, it is necessary to describe the error characteristic at the beginning of such an error. This allows to select a matching recovery strategy (cf. Chap. 5).

Since the focus is on assembly by mating, movements around or along a mating axis are considered. Therefore, the mating axis A_M needs to be part of the state attributes. A reference coordinate system $\mathbf{F}_{Ref}$ is needed for describing rotational or transitional difference between an error and a normal assembly operation. In Sec. 5.3, various ways for selecting an assembly strategy are presented. All require that a preferred selection criteria (cf. Sec. 5.3) is defined by a domain expert. The reason for this is that a selection criterion depends on the nature of an error. By the fact that the domain expert defines it, he can take into account his additional knowledge about the specifics of an error. In addition, a measure for determining the performance of a recovery strategy is needed in Sec. 5.5. It is called PC_{Pref} and offers the possibility to run the fusion approach for optimizing a strategy database. But this performance measure depends on the error. For

this reason, it needs to be defined by the domain expert and it is added to the attributes of a state.

An overview of all state attributes is given by Tab. 4.1. With these state attributes it is possible to select one out of a number of recovery strategies belonging to the same error class (cf. Fig. 1.4). It should be noted that these attributes are not needed for pure assembly process monitoring. They are only used for recovering from errors during assembly.

Table 4.1.: State properties needed for recovering from errors during assembly operations.

Symbol	Description
A_M	Movement along or around the mating axis A_M is possible during mating operation.
$\mathbf{F}_{Ref}$	The z-axis of reference coordinate system $\mathbf{F}_{Ref}$ is equal to A_M by definition. Additionally, an origin on A_M and a respective x- or y-axis uniquely specify the full $\mathbf{F}_{Ref}$.
SC_{Pref}	Selection criterion SC_{Pref} which is used, if different criteria are applicable.
PC_{Pref}	The performance criterion PC_{Pref} is used for rating of different recovery strategies.

4.2. Handling of Uncertainties

While decomposing complex assembly tasks, process uncertainties can occur due to variations between different datasets and during automatic classification (cf. Sec. 4.5) of input data. So for a robust application of the *HD* during data analysis handling state probabilities needs to be introduced. Also multiple domain experts can be involved in the segmentation phase (cf. Sec. 4.3). This can lead to the problem that various experts use slightly different definitions of the conditions C_U, so that there are uncertainties in the definition of a states $\mathbf{S}$. As a consequence, the transition between different states is diffuse. This aspect is shown in Fig. 4.3.

Each state $\mathbf{S}_{i,k}$, is extended by a state probability $p_{\mathbf{S}_{i,k}}$ for extending the already mentioned definition of the *HD*. For all state probabilities p of a hierarchical level L_k with n_k states $\mathbf{S}_{i,k}, i \in \{1, ..., n_k\}$, including the special states it takes effect $\sum_{i=1}^{n} p_{\mathbf{S}_i} = 1$. Because the conditions $C_{i,j,k}$, describe transitions of probable states, boolean logic regarding these conditions is no longer usable. Instead, a probability p is added to the user defined part C_U of each condition. So p for C_U in $C_{i,j,k}$, is equal to p of $\mathbf{S}_{j,k+1}$. Since C_R is used to

encode preconditions, the probability p of C_R is zero until the condition is fulfilled in order to represent a binary decision.

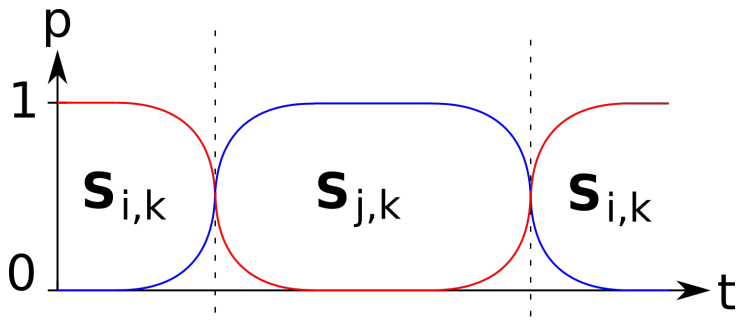


Figure 4.3.: Exemplary state transition from $\mathbf{S}_{i,k}$ to $\mathbf{S}_{j,k}$ and back to $\mathbf{S}_{i,k}$. The probability of each state for being the active one as a evolution over time is shown.

4.3. Domain Expert's Workflow

A domain expert has to define states and conditions. Also, a manual segmentation needs to be done by the domain expert for generating training data. By using this data, automatic classifiers for handling a larger dataset can be trained. The definition process is an iterative one that can be supported by the previously trained classifiers. The entire procedure is depicted in Fig. 4.4 and the single steps are described during the following subsections.

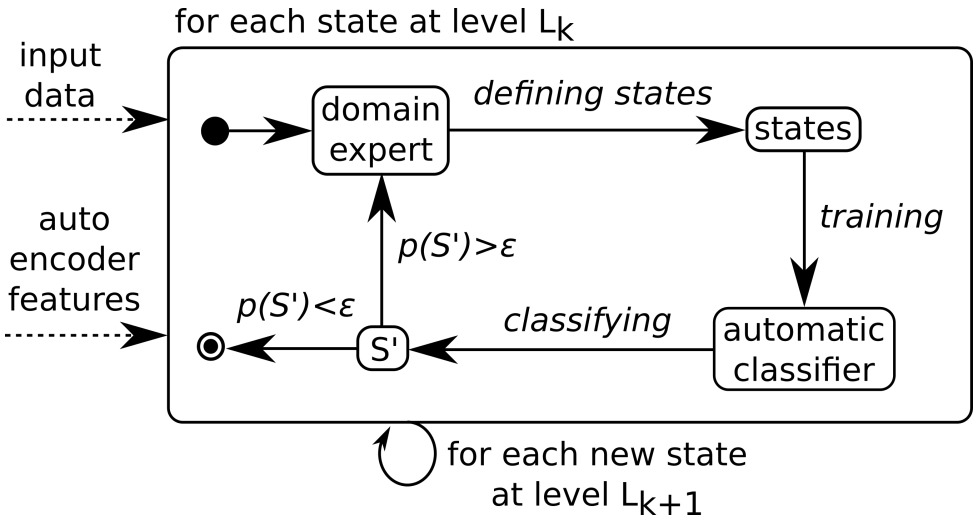


Figure 4.4.: State chart showing the decomposition workflow.

When focusing on industrial assembly tasks and the identification of human strategies, it is practical if the input data contains the pose of the workpiece and also the acting forces and torques. This data can simply be recorded by utilizing an industrial manipulator and applying *PbD*. However, the presented approach is not limited to this data capturing method. Another way is to place the workpiece on top of a force-torque sensor. By using a motion tracking system, also data from assembly tasks executed by hand can be processed.

The domain expert's role is to generate training data for training automatic classifiers (cf. Sec. 4.5). Therefore, manual segmentation of the input data is required. During this phase, the domain expert is not limited to make decisions only due to the input data. Also meaningful features, calculated by stacked auto-encoders (cf. Sec. 4.5), can be used for decision-making. Typically, the expert starts by regarding a complete dataset which represents the first level L_2 (one below the root at L_1) of the decomposition tree. After defining states for this level, the expert has to decide whether one of these states needs further decomposition. Here, the probability of the special states $\mathbf{S'}_{i,k}$ is helpful because the value $p(\mathbf{S'}_{i,k})$ is high if the defined states are not sufficiently explicit to describe the input data. So, the expert has to define a value ϵ and test whether the state definition is sufficiently expressive by evaluating it holds $p(\mathbf{S'}_{i,k}) > \epsilon$. If it is insufficient, further states need to be defined for creating an expressive *HD*. Depth and structure of this decomposition are determined by task specific knowledge of the domain expert because interesting ranges of the input dataset are segmented in more detail than others.

4.4. Feature Vector for Automatic Classification

The output of a manual segmentation phase performed by the domain expert is used for training an automatic classifier. Therefore, it is necessary to calculate a feature vector. Statistical features are suitable for representing the evolution of a signal over a time interval. But their use requires the input data to be split up into sets so that these features can be calculated for each set. This problem is addressed by using a rectangular window for feature extraction, as shown in Fig. 4.5. An element j of the feature vector, corresponding to sample i of the data, is calculated by using all data samples in $\{i - w_{pre}, i + w_{post}\}, w_{pre}, w_{post} \in \mathbb{N}_0$. Using such a window causes that the feature vector has w_{pre} elements less at the beginning and w_{post} elements less at the end than the data vector. Splitting the window into a part $\{w_{pre}\}$ before sample i and one $\{w_{post}\}$ after it allows to use an asymmetrical window. This is an important aspect if the decomposition is applied for online error detection or prediction (cf. Sec. 4.6).

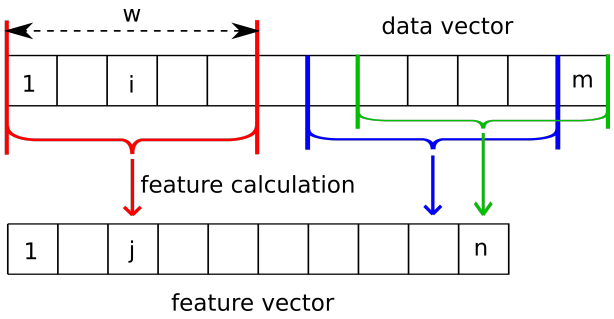
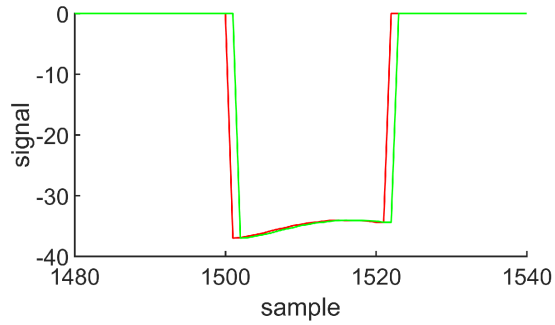


Figure 4.5.: Extraction of n feature samples from m data samples by using a moving window w. Three different windows (red, green and blue) are used for demonstrating the extraction w.r.t. to three sets of data.

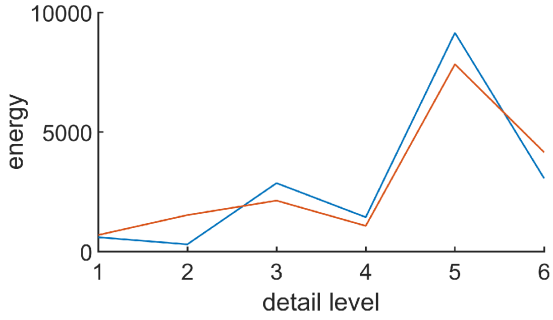
Apart from using statistical features, the Fourier transform can be used to calculate spectral features. But it is more meaningful to use a time-frequency representation. Various approaches from different research areas recommend using wavelet coefficients or wavelet based time spectral energy distribution [24, 36, 82, 40].

By Paseval's theorem, the energy of a signal is related to the energy of each subband. Therefore, the energy distribution should remain unchanged if the signal is shifted in time. But the wavelet transform is not invariant w.r.t. time shifting [41]. As an example, Fig. 4.6(a) shows two signals which are shifted by one discrete time step. After computing the wavelet energy distribution, there is a significant difference between both energy distributions, as shown in Fig. 4.6(b). Therefore, the standard wavelet transform is not the best choice for pattern recognition based on the energy distribution.

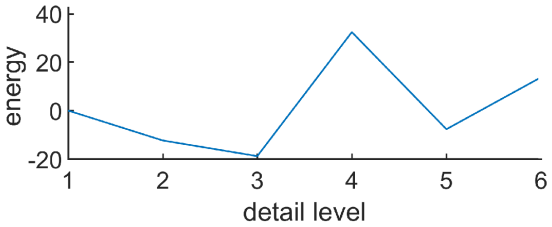
Instead, the *Dual-Tree Complex Wavelet Transform* (DTCWT) should be used [77]. For the same two signals, this transform causes a considerably smaller difference between both energy distributions. Fig. 4.6(c) and Fig. 4.6(d) show the differences in energy distribution of both transforms w.r.t. to the already mentioned signals. But it should be noted that the dual-tree complex wavelet transform causes an error if a signal is reconstructed.



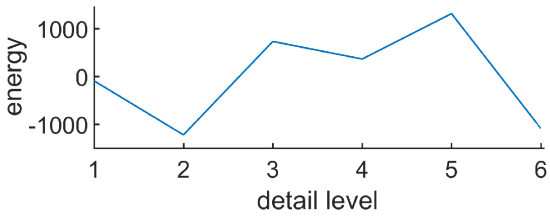
(a) Both signals (red and green) are similar. The only difference is that the green one is shifted by one discrete timestamp.



(b) Energy Distributions of both signals by using the Wavelet Transform.



(c) DTCWT: Difference in energy distribution between both signals.



(d) Wavelet Transform: Difference in energy distribution between both signals.

Figure 4.6.: Effects of a signal which is shifted in time onto wavelet based energy distributions.

The *HD* is intended as a formal representation of industrial assembly tasks. Because of this, data which is typically available during *PbD* is used as an input. An overview of the inputs used here is given by Tab. 4.2. It should be noted that the presented approach is not limited to these inputs and others can also be used.

Table 4.2.: Input data used for calculating a feature vector.

Position of the end-effector: $\vec{p}$
Orientation, represented as a quaternion: q
Acting force vector: $\vec{F}$
Acting torque vector: $\vec{M}$

The features mentioned in Tab. 4.3 have been selected because they describe the statistics and the evolution of a signal. Also, a time-frequency representation is generated by using the DTCWT. All following features can be calculated for input data $x \in \{\vec{p}, q, \vec{F}, \vec{M}\}$. If the mentioned window is used, this is indicated by the window function $w()$.

Table 4.3.: Overview of the applied features.

Standard deviation: $\sigma(w(x))$
Variance: $\sigma^2(w(x))$
Mean: $\overline{w(x)}$
Difference: $x_i - x_{i-1}$
Pitch of linear approximation w.r.t. $w(x)$
Range: $max(w(x)) - min(w(x))$
Inter quartile distance: $Q_{.75} - Q_{.25}$ w.r.t. $w()$
Mean deviation from median: $\frac{1}{n}\sum_{i=1}^{n} \lvert x_i - \tilde{x}\rvert$ w.r.t. $w()$
Vector direction: $\frac{\vec{x}}{\lvert\vec{x}\rvert}$
Vector length: $\lvert\vec{x}\rvert$
Difference of directions: $\frac{\vec{x}_i}{\lvert\vec{x}_i\rvert} - \frac{\vec{x}_{i-1}}{\lvert\vec{x}_{i-1}\rvert}$
DTCWT energy distribution for n levels: $\frac{E_i(w(x))}{E_{total}(x(w))}, i = \{0, n\}, n \in \mathbb{N}$
DTCWT coefficients of x

4.5. Automatic Classification

Due to the high number of features, a reduction of the feature space is needed for decreasing the complexity of training automatic classifiers. A simple approach is to perform a feature selection by using clustering techniques. Therefore, each feature is clustered and only features with a distance between their cluster centres greater than a threshold are used [40]. The disadvantage of this approach is that not all available information is used because it is only a feature selection approach. Therefore, the Principal Components Analysis (PCA) is typically used for feature reduction [48]. But this approach cannot describe complex and highly non-linear correlations between several features. However, this is achieved by using stacked auto-encoders as described by [9].

Here, a deep learning approach based on neural networks for solving the classification task is used. The network architecture is build up by stacked auto-encoders for feature reduction and a softmax layer for classification. Each layer of the resulting network is trained on its own and at the end of the training a fine tuning of the whole network is performed. This corresponds to the classical approach for training a deep neural network [26].

Such a network is used to classify the input data into states at the same hierarchical level and the network returns the state probability for each of these states. Different networks are used on different hierarchical levels, as shown in Fig. 4.7. If states on the same level L_k are children of different states at level L_{k-1}, a single network is used for each branch.

For example, if a state $\mathbf{S}_{i,k}$ is connected by conditions with a set of states $\mathbf{S}_{i,k+1}$, then a single network is used for classification. But if two states $\mathbf{S}_{i,k}$ and $\mathbf{S}_{j,k}$ of the same level L_k are connected with two different sets at level L_{k+1}, then a separate network is used for each of this set. The usage of one network for both sets would increase the complexity of the classification problem unnecessarily. Because of the *HD*'s definition, states out of the set connected to $\mathbf{S}_{i,k}$ are not possible if the active state at level L_{k+1} is connected to $\mathbf{S}_{j,k}$.

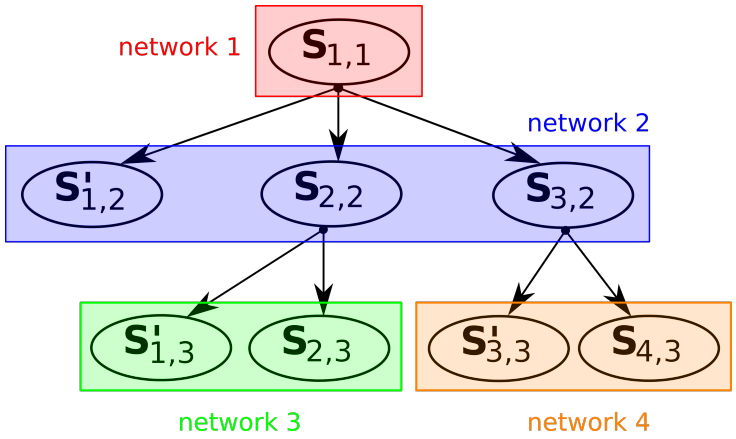


Figure 4.7.: For classification, at least one network has to be used per hierarchical level. Multiple networks can be used on the same level L_k if the states of this level are children of different states at level L_{k-1}.

4.6. Application of the Hierarchical Decomposition

In literature, a simple mating task is typically used instead of a complete assembly task. A well-known example from the industrial context is the *Peg-in-Hole* task. Here a similar task is used. The reason for this is that an extensive user study on *KG* in assembly operations has been previously executed (cf. Chap. 3). So, it is possible to reuse the collected data for exemplary application of the *HD*. The user study included examples of varying difficulty. A detailed description of each example and of the experimental setup can be found in Chap. 3 and in [58]. Instead of a *Peg-in-Hole* task the *Spline Shaft* task is used here. The reason is that due to the larger socket and due to the reduced *DoF* more errors have been observed for this task. Hence, *HD*'s advantages for online error detection and prediction and also for giving a detailed description of problems which occur during task execution would be less obvious. Both workpieces, the spline shaft and the corresponding socket, are depicted in Fig. 4.8. Engineering drawings can be found in Appendix A.2.

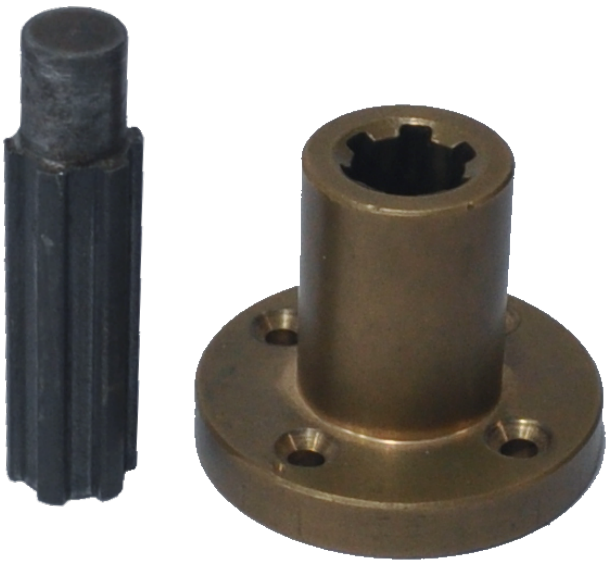


Figure 4.8.: Spline shaft and socket used as a more meaningful example compared to the typical *Peg-in-Hole* widely used in literature.

For the example *Spline Shaft*, typical problems during mating have in common, that the z-position stays constant during the problematic areas. This can have different causes like mechanical jamming, friction or other physical effects or are even intentional breaks during assembly. Typically, multiple errors occur while assembling. An example is given by Fig. 4.9. This figure includes above average problematic areas. All 397 spline shaft datasets of the mentioned user study include 1134 errors. Just as the average score is about 2.96 errors per dataset, it is important to have an assembly monitoring tool and also a possibility to recover from such errors if they occur during automatic assembly.

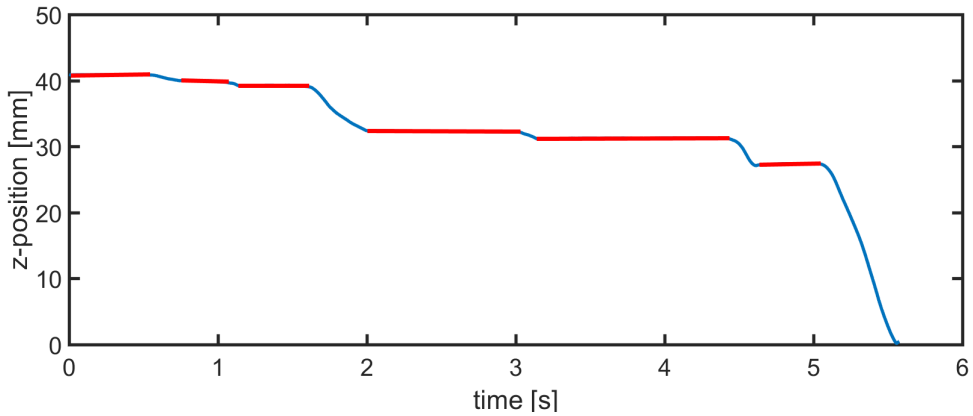


Figure 4.9.: Plot of the spline shaft's z-position over time. A constant z-position (red-colored ares) indicates the occurrence of an error during assembly. The assembly operation starts at a z-position of 40 mm (height of the socket) and ends at 0 mm when the spline shaft reaches the bottom end of the socket.

4.6.1. Simplifications and Parametrization

Here, the focus is on online error detection and prediction. Description and classification of errors which occur during assembly is also of particular interest. Hence, it is only necessary to consider a subset of the available input data (cf. Tab. 4.2). Due to spline shaft's geometry and that of its socket, shown by Fig. 4.8, the spline shaft has only one *DoF* during assembly. No orientation information is used here. The remaining *DoF* is a translation along the longitudinal axis of the socket. This axis is called the z-axis and hence, Fig. 4.9 is limited to the z-position of the spline shaft. Other translational movement is impossible because the mating operation is spatial limited by the socket of the spline shaft.

Because of these simplifications, the size of the feature vector, which is used during the following examples, is reduced. In contrast to Tab. 4.2 the input data only consist of p_z, $\vec{F}$ and $\vec{M}$.

The *HD* of the *Spline Shaft* experiment is limited to four hierarchical levels. Otherwise, the decomposition would become too complex for a short example. In addition, information on training automatic classifiers and on their performance is given. After identifying a problematic segment of the *Spline Shaft* task, a detailed view of this segment is presented.

For training the automatic classifiers corresponding to the specified states, a feature vector containing 160 features was calculated. The number of used features was reduced due to

the already mentioned simplifications. For calculating the feature vector, a symmetric window with $w_{pre} = w_{post} = 17$ was used. The DTCWT was applied with 7 levels. Thus, the narrowest and lowest frequency band is 15.625 Hz wide. This is caused by using a sampling frequency of 2 kHz. This DTCWT frequency resolution fits to human arm and finger motion. That is an important point because the goal of this decomposition is to describe the strategies of a human during interaction with the robot. Typical intended human arm or finger motions can reach a frequency up to 12 Hz [51]. Therefore, it is possible to distinguish between intended human movements and other effects. The used deep network consists of two stacked auto-encoders with 110 and 13 outputs and a softmax layer for classification. Two different datasets were used for training and evaluation. Each included 75 task executions.

4.6.2. *HD* of the *Spline Shaft* Assembly Task

A graphical representation of the *HD* is given by Fig. 4.10. Tab. 4.4 contains the description and the classification performance of each state. It should be noted that the shown performance scores were calculated by using only the samples within the windowed area. Information e.g. which state was previously active or whether a state transition according to the *HD* is even possible was not used. Such approaches can improve the performance scores.

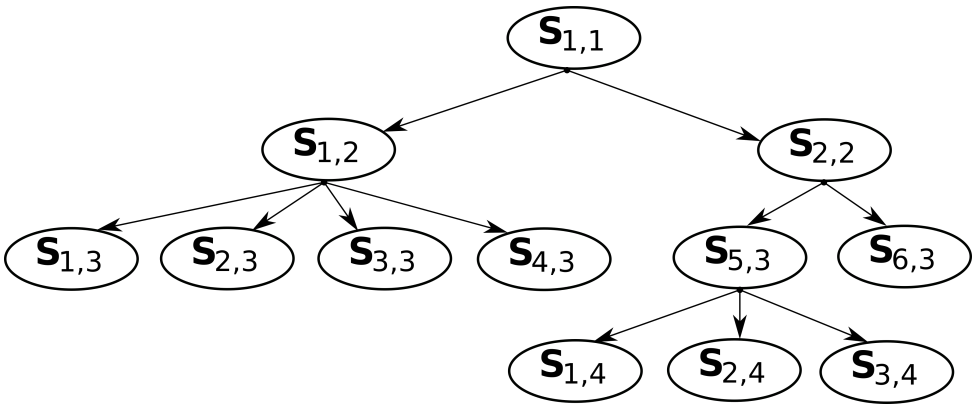


Figure 4.10.: *HD* of the *Spline Shaft* task without $\mathbf{S}'_{j,k}$ states. A description of each state is given in Tab. 4.4.

Looking at the decomposition tree, the first level consists of the entire task because it is the root. Employing the above mentioned simplifications, the task progress can be described by the spline shaft's z-position. If there is no progress the z-position stays constant and when that value is equal to zero, the task is completed. Because of this, the domain

expert has chosen a variation of the z-position for defining two conditions. The first one, $C_{1,1,1}$, means that there is a changing z-position and hence a mating progress. The second condition $C_{1,2,1}$ defines that there is no mating progress.

Table 4.4.: *HD* of exemplary considered the *Spline shaft* task.

Level	Element	Description	Classification performance
L_1	$\mathbf{S}_{1,1}$	$\ll$*root*$\gg$	100%
L_2	$\mathbf{S}_{1,2}$	$\ll$*mating*$\gg$	93.9%
	$\mathbf{S}_{2,2}$	$\ll$*notMating*$\gg$	93.9%
L_3	$\mathbf{S}_{1,3}$	$\ll$*mating*$\gg$	91.1%
	$\mathbf{S}_{2,3}$	$\ll$*preBlocking*$\gg$	89.2%
	$\mathbf{S}_{3,3}$	$\ll$*firstContact*$\gg$	87.7%
	$\mathbf{S}_{4,3}$	$\ll$*finalMating*$\gg$	99.8%
	$\mathbf{S}_{5,3}$	$\ll$*blocking*$\gg$	90.3%
	$\mathbf{S}_{6,3}$	$\ll$*break*$\gg$	93.7%
L_4	$\mathbf{S}_{1,4}$	$\ll$*rising*$\gg$	96.1%
	$\mathbf{S}_{2,4}$	$\ll$*falling*$\gg$	95.7%
	$\mathbf{S}_{3,4}$	$\ll$*constant*$\gg$	93.4%

All substates of $\mathbf{S}_{1,2}$ describe the actual mating process. $\mathbf{S}_{3,3}$ describes the first contact between the spline shaft and its socket. C_U of $C_{1,3,2}$ is defined so that the acting forces and torques are above the sensor noise level and that there is a nearly constant z-position. C_R demands that both workpieces have not been in contact before. The state $\mathbf{S}_{3,3}$ or $\ll$*firstContact*$\gg$ describes how the spline shaft is inserted into the socket. The mating task's end is represented by the state $\mathbf{S}_{4,3}$ or $\ll$*finalMating*$\gg$. For being in this state, $C_{1,4,2}$ requires a high z-force peak and after that a constant z-position. Another substate is $\mathbf{S}_{2,3}$ or $\ll$*preBlocking*$\gg$ and this substate is used for online error prediction. The last substate is $\mathbf{S}_{1,3}$ which is named $\ll$*mating*$\gg$ again. Thus, this state describes the mating progress at level L_3.

The domain expert splits $\mathbf{S}_{2,2}$ into two states. The first is named $\mathbf{S}_{6,3}$ or $\ll$*break*$\gg$. For being at this state, $C_{2,6,2}$ has to be fulfilled and this condition requires a constant z-position, forces and torques. The domain expert defined that a constant z-position and changing of forces or torques are equal to the condition $C_{2,5,2}$. The corresponding state is named $S_{5,3}$ or $\ll$*blocking*$\gg$. This is a very interesting state because it includes a problem during task execution. On higher decomposition levels, the substates of this state also describe how the problem is handled in order to solve it. But on higher levels, the domain expert has to be aware of the maximum human movement frequency in order to identify intended human actions. If a state is only maintained for a short time, it cannot represent an intended movement. So the domain expert has to define a C_R according to

the maximum movement frequency of a human.

Since the human understandable error description and the successful error handling is of special interest, a decomposition of an exemplary problem was given. The presented experiment includes the state $\mathbf{S}_{5,3}$ or $\ll$*blocking*$\gg$. Such an exemplary and simple state with its substates is shown in Fig. 4.11. In order to simplify the visualization, only the force and not the torque vector are shown. Only one decomposition level of this state is shown. As determined by the condition $C_{2,5,2}$, the z-position for the duration of the entire state is constant. In Fig. 4.11, F_z has a negative sign because the coordinate system of the force-torque sensor is oriented opposed to the mating direction.

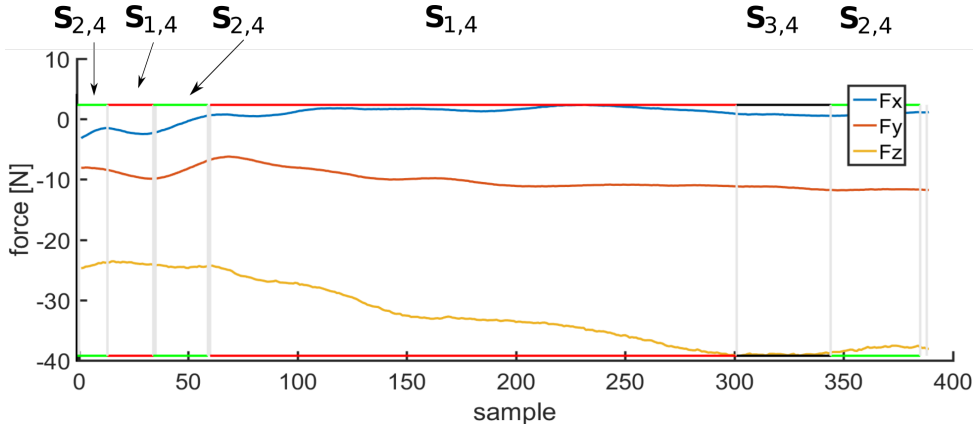


Figure 4.11.: Representation of the force evolution (F_x: blue, F_y: red, F_z: yellow) over the samples during $S_{5,3}$ or also $\ll$*blocking*$\gg$. The figure shows the whole state $\mathbf{S}_{5,3}$. Red, green and black lines at top and bottom are used to label substates of $\mathbf{S}_{5,3}$.

4.6.3. Composition of an Assembly Error

By extending the decomposition with an additional hierarchical level L_4, it is possible to describe $\mathbf{S}_{5,3}$ or $\ll$*blocking*$\gg$ state with more detail. At level L_4 three more states are defined. The first one is $\mathbf{S}_{1,4}$ and the associated condition is $C_{5,1,3}$. The domain expert defines that $C_{5,1,3}$ is equal to a rising magnitude of the force vector. The condition $C_{5,2,3}$ describes the state transition to $\mathbf{S}_{2,4}$ and is equal to a falling magnitude of the force vector. $\mathbf{S}_{3,4}$ is the last substate of $\mathbf{S}_{5,3}$ and the condition for state transition is $C_{5,3,3}$. This is equal to a constant magnitude of the force vector. All states are labelled in Fig. 4.11.

This figure also consists of a temporally ordered sequence of L_4 states. So it is possible to describe how the force vector has changed by viewing at the changing conditions. At the beginning, the magnitude of the vector was decreased. Then it was increased, decreased,

increased, constant and at the end it was falling. Only the fourth element of this chain consists of enough samples so that it could be an intended human motion. The other elements can be caused by physical effects like stick-slip friction between the spline shaft and the socket while a nearly constant force pushes the spline shaft. This aspect has to be keep in mind if such a chain is used for automated robot program generation.

A statement regarding the percentage composition of $\mathbf{S}_{5,3}$ is given by Tab. 4.5. In addition, information on the occurrence of a substate at the beginning or end of $\mathbf{S}_{5,3}$ is given.

Table 4.5.: Percentage composition of $\mathbf{S}_{5,3}$ or $\ll$*blocking*$\gg$ state w.r.t. to the evaluation set

State	Percentage of occurrence		
	Overall	Beginning	End
$\ll\mathbf{S}_{1,4}\gg$	36.4	14.3	14.1
$\ll\mathbf{S}_{2,4}\gg$	47.3	80.9	71.4
$\ll\mathbf{S}_{3,4}\gg$	16.3	4.8	14.5

4.6.4. Online Error Detection and Prediction

A simple approach for online error detection during task execution is to define error states so that they are elements of the hierarchical decomposition. The $\ll$*blocking*$\gg$ state from Tab. 4.4 is an example of such an error state. During such a state, the task execution is disturbed and a special error handling is necessary to allow a further progress of the task.

By applying the trained automatic classifiers in parallel to the task execution to data which is sampled at the same time, it is possible to decide if the current state is an error state or not. Avoiding error states is better because the workpieces or the equipment can be damaged during such an error state. Using the *Spline shaft* example, it is possible to demonstrate how a pre-error state can be defined and used for error prediction. This state makes it possible to avoid error states and for preventing potential hardware damage or for enabling an early error prevention.

Force-torque data of the *Spline Shaft* experiments was sampled with a frequency of 2 kHz. A domain expert observed a clear oscillation of measured forces and torques before most $\ll$*blocking*$\gg$ state. Depending on the velocity of task execution, oscillating forces and torques can be last for up to 150 samples. So it was possible to define a $\ll$*preBlocking*$\gg$ state and to detect this state with a success rate of 89.2% as shown in Tab. 4.4.

In contrast to the classification of already captured data, the window function for feature generation can only use sampled data. Here, w_{pre} was set to 34 and w_{post} to 0. Thus, it takes 17 ms to reliable detect a pre-error state. But since the $\ll$*preBlocking*$\gg$ state takes up to 75 ms there is still time left to react. Such a reaction could be based on typical recovery strategies identified by decomposing $\ll$*blocking*$\gg$ states.

Chapter 5

Recovery Strategy

This chapter defines a recovery strategy w.r.t. an industrial assembly scenario. Specific requirements for a simple application are also presented. Basis of the chapter are previous publications [61, 62]. In addition, criteria for describing assembly errors are introduced. They are also used for selecting a recovery strategy. So, it is highly important that a selected strategy matches a given assembly error. The strategy representation and the selection criteria are invariant against rotation or translation of the considered workpieces. So, generalization is enabled and the number of required human interaction is minimized. Note that the recovery strategies are task specific, but robot independent. Therefore, they can be reused for the same task but also by other assembly systems or robots. The shop-floor workers' task specific knowledge and experience is exploited during the demonstration phase and encoded inside the strategy, which is an important feature.

An optimization of the strategy selection process is also presented so that all information regarding an assembly error is used. Due to the fact that human demonstrations are rarely optimal, it is shown how different strategies can be fused together for creating new ones with better properties.

5.1. Requirements of a Recovery Strategy

Two different problem classes are decisive for the recovery strategy definition. For the first class (cf. Sec. 5.1.1) it is typical that little information regarding an assembly error is known. The decisive factor here is the poor observability not only on a microscopic level. Within this work it was not possible to determine human behaviour and intentions. Thus, the influence of these points on the success or failure of a strategy is unclear. So, the second problem class (cf. Sec. 5.1.2) is based on these points.

5.1.1. Lack of Observability Limits Modelling

Especially for assembly tasks, the problem arises that their executions are difficult to observe. Optical metrology is unsuitable due to frequent occlusions. With regard to industrial assembly tasks, the use of additional external measuring technology is undesirable. Thus, in automated assembly, information regarding the workpiece's pose can only be determined based on the robot's pose. Acting forces can be either measured by a force sensor or be derived from the robot's joint torques by using the dynamic model. Therefore, the exact contact situation of the workpieces is unknown. Especially on a microscopic scale, information about the state of the workpiece's surface, the blocked geometries or the acting physical effects, e.g. friction, is needed. Without this information, errors cannot be classified based on their causes. However, such a classification is needed for describing the error recovery by using stochastically models like Hidden-Markov-Models (*HMM*) or Gaussian-Mixture-Models. If a classification would be possible, a *HMM* could be trained for each class. This would mitigate the problem that training data is needed for each *HMM*. Collecting this data is expensive because first the error needs to be exactly reproduced during the assembly operation. Second, the corresponding recoveries need to be manually demonstrated.

One possible approach is to roughly describe errors (cf. Sec. 5.3) by using all available sensor data. So, a matching recovery strategy can be selected. Subsequently, the recovery can be performed by manipulating the workpiece in exact the same way it was done in the previous demonstration. There can be no assurance that the recovery will be successful because the difference between error causes is unknown. However, in case of a failure it is possible to try another recovery strategy w.r.t. the varied error.

5.1.2. Influences of Human Behaviour

When human operators are asked to demonstrate recoveries from an error (cf. Sec. 1.3), it is not possible to determine the intended purpose of an action. The problem is that the human intentions are not recognizable in the experimental setups used in the user studies presented in Chap. 2 and Chap. 3. Within these studies, only the robot's pose was sampled. Thus, only human hand motion could be estimated. Without recognizing human intentions it is not possible to say whether a demonstration includes one or more strategies. It might be the case that a human noticed, that the current strategy which is applied, does not work. Therefore, the human might switch to another one. So, when using demonstrations for recovery, always a complete demonstration must be used. Otherwise, it cannot be guaranteed that every action necessary is included. Thus, also removing unnecessary or redundant parts is not possible.

5.2. Definition of a Recovery Strategy

Assembly errors often appear seemingly similar on a macroscopic level but small variations regarding the starting conditions lead to differences. Therefore, a recovery strategy always addresses a specific error. The sampling of enough recovery demonstrations for covering the variance regarding the starting conditions implies significant effort. A suitable representation is needed to exploit geometric invariance and increase re-usability, respectively. Therefore, the overall effort can be reduced. Representation in relative coordinates is e.g. frequently used in learning by demonstration approaches [55]. A strategy contains relative movements w.r.t. to the end-effector's pose at the moment an error occurs. As a result, rotation and translation invariance is achieved. Considering for example the classic *peg-in-hole* assembly task, the movement of a peg looks different from the view of an external observer if the starting pose is rotated around the longitudinal axis of the peg. However, it is the same movement w.r.t. the starting pose. A graphic example is given in Fig. 5.1.

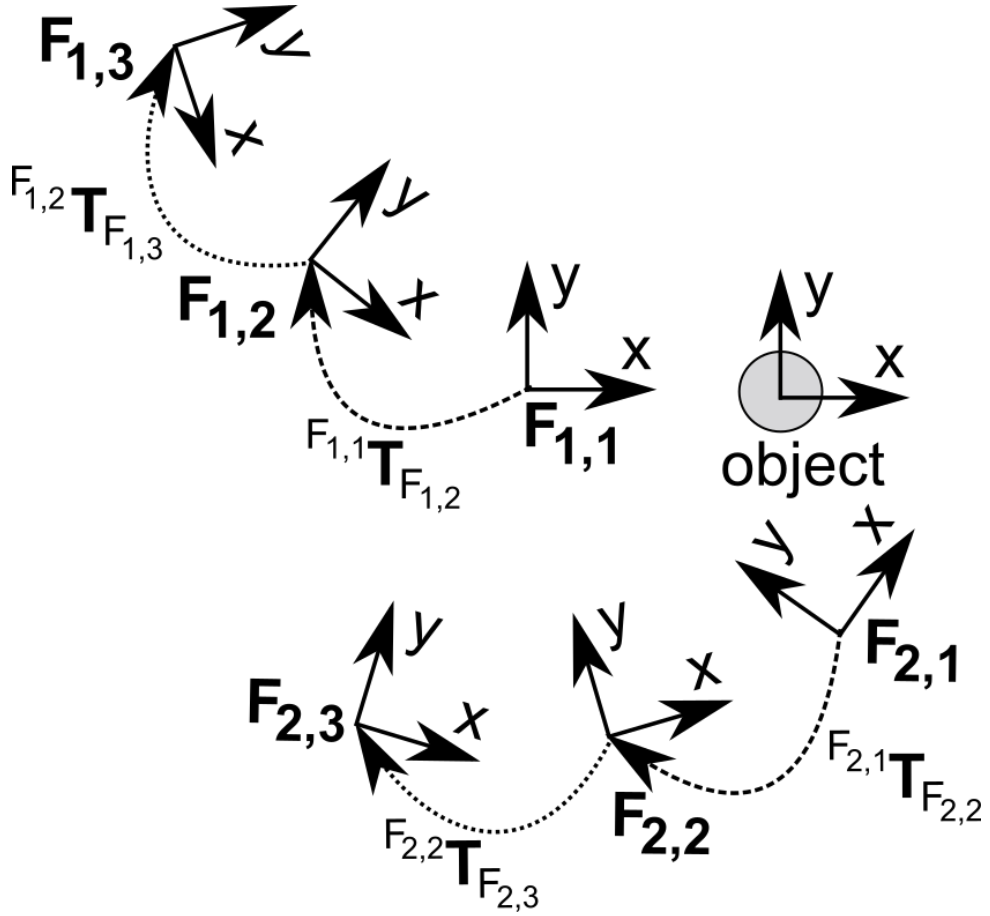


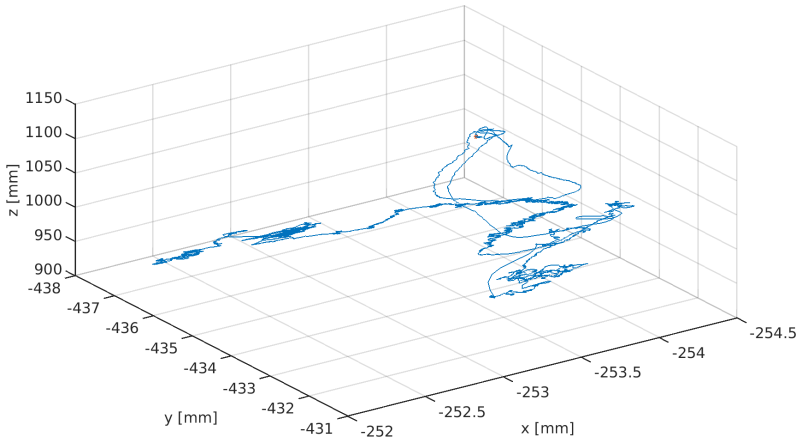
Figure 5.1.: 2D example of how the same motion looks different if its viewed from an external point like the object. But regarding the first frame $F_{\{1,2\},1}$ it is obviously the same motion.

On the other hand, it is not possible to achieve such an invariance regarding the velocity profile. The reason is that friction is an important feature of the contact between both workpieces during motion. Their relative velocity is an important factor in friction models e.g. the well-known LuGre model [38] or more complex ones [32]. Here, a fixed rate is employed for sampling human demonstrations of recovery strategies. So, the same velocity profile is reproduced if a strategy is used w.r.t. the sampling rate. Re-sampling can be applied when necessary in reproducing it.

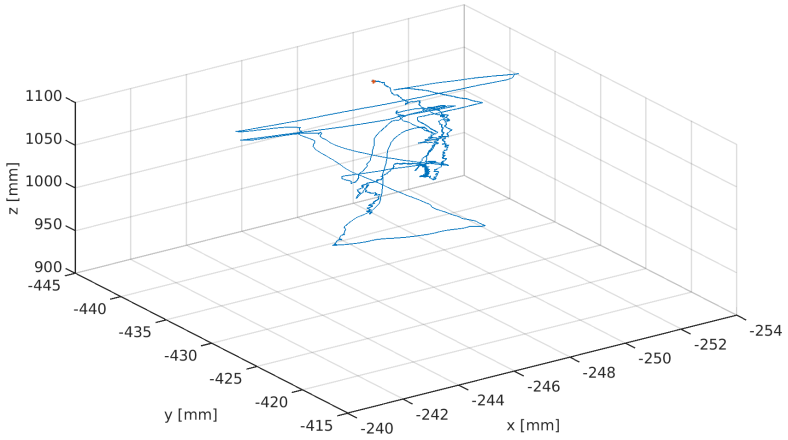
Taking these requirements and limitations into account, there are two ways to present a recovery strategy. Both require a workpiece-centred reference frame $\mathbf{F}_0$ for describing the end-effector motion. It is given by the end-effector pose at that point in time when the error is detected. If a recovery strategy is demonstrated $n, n \in \mathbb{N}$, end-effector poses $\mathbf{F}_i, i \in [0, n-1], i \in \mathbb{N}_0$ are sampled. Both ways of representing a recovery strategy differ

in terms which reference frame is used for describing the movement. One possibility is to use $\mathbf{F}_0$ as reference for all other sampled frames so that a strategy consists of $n-1$ transformations ${}^{F_0}\mathbf{T}_{F_j}, j \in [1, n-1], j \in \mathbb{N}$. This set of transformations is ordered in the way that $\mathbf{F}_j$ is sorted w.r.t. an ascending order of the index j. The other possibility of representing a recovery strategy is that the transformations always describe the motion between the previous and the current pose w.r.t. the previous one. In this case a strategy is also an ordered set of the transformations ${}^{F_{j-1}}\mathbf{T}_{F_j}$. However, this way of representation has the disadvantage that it leads to an accumulation of errors. That is why the first way is preferred. Nevertheless, both are invariant against rotation or translation of $\mathbf{F}_0$ and both reproduce the same velocity profile as during the sampling of the recovery strategy.

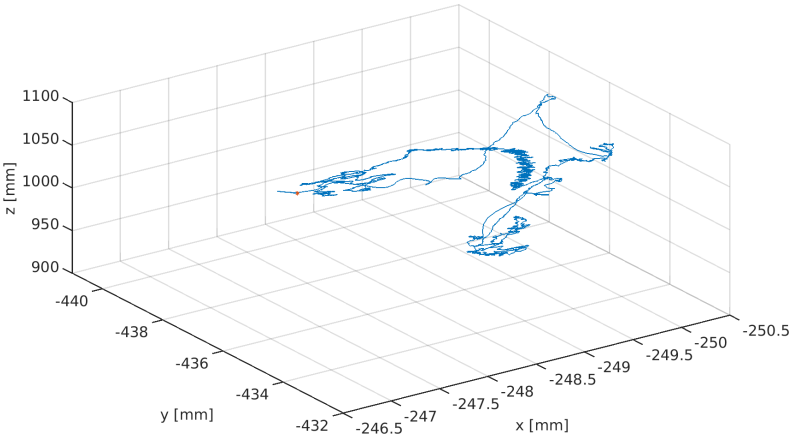
An example, how such a strategy might look like, is shown in Fig. 5.2, where three different strategies are shown. The selection made was created randomly. All strategies have been successfully used in a user study (cf. Chap. 2) for recovering from errors w.r.t. *Pleuel - Top*. Therefore, it is the goal to show the small end-effector motions during recovery. These movements are made possible by the minimal compliance of the experimental setup.



(a) Example No. 1.



(b) Example No. 2.



(c) Example No. 3.

Figure 5.2.: Three examples of how a recovery strategy looks like are depicted. The shown strategies have been randomly selected of out a database for the assembly task *Pleuel - Top*.

5.3. Recovery Strategy Selection

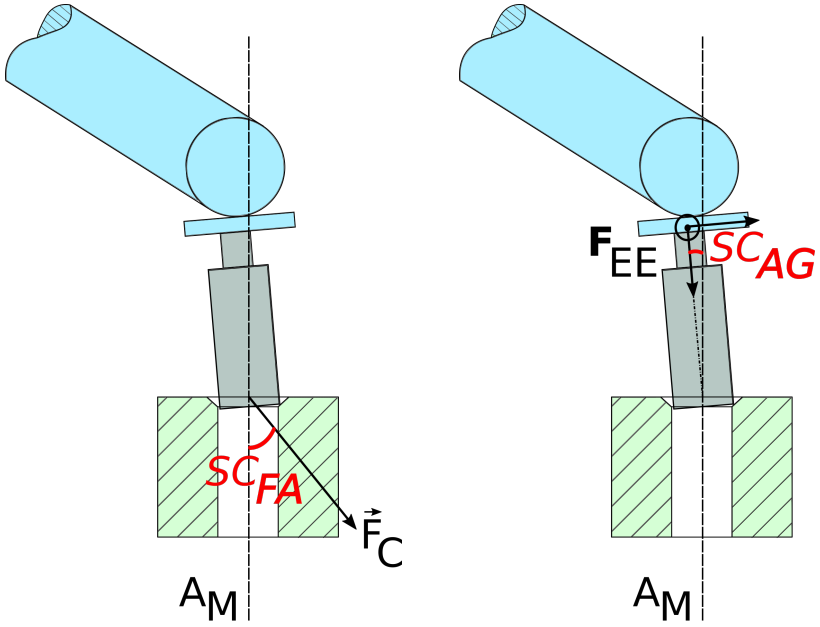
The microscopic state of an assembly operation is unknown if an error occurs. Therefore, the problem arises how to select a suitable recovery strategy. Criteria which can be used for describing a situation and thus for selecting a strategy are called selection criteria SC. Here, the focus is on assembly by mating according to *DIN 8593-1:2003-09* which specifies different ways of mating based on translational and rotational motions. The strategy representation (cf. Sec. 5.2) and also the SC need to be invariant against rotation and translation as far as possible. In order to evaluate the SC the acting contact force vector $\vec{F}_C$, the current approach vector of the end-effector coordinate system $\mathbf{F}_{EE}$ and $\mathbf{F}_{Ref}$ for describing rotational motions w.r.t. the axis of $\mathbf{F}_{Ref}$ are used.

SC used for describing assembly errors:

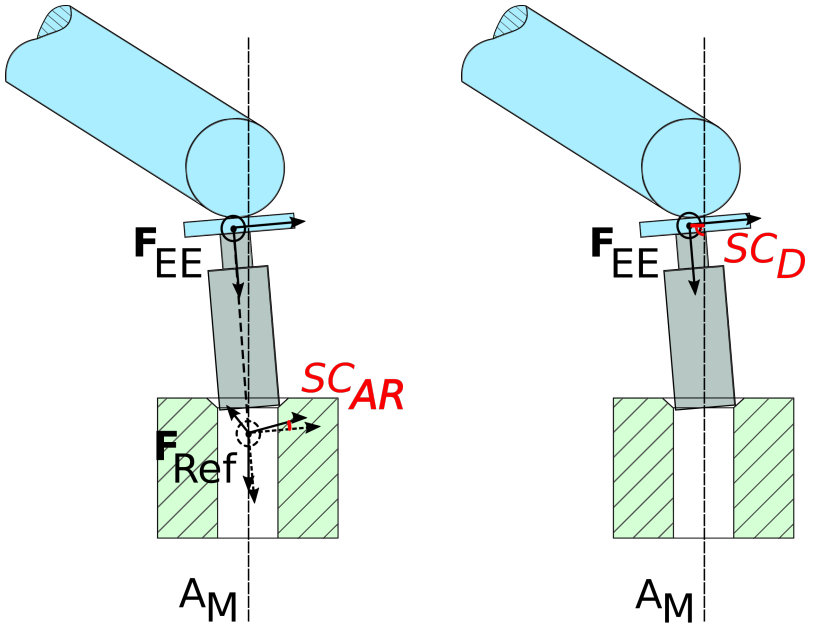
- SC_{FA}: Angle between the mating axis A_M and the contact force vector $\vec{F}_C$.
- SC_{AG}: Angle between the approach vector of $\mathbf{F}_{EE}$ and that of $\mathbf{F}_{Ref}$.
- SC_{AR}: Angle between the $x-$ or $y-$axis of $\mathbf{F}_{EE}$ and that of $\mathbf{F}_{Ref}$.
- SC_D: Shortest euclidean distance between the origin of $\mathbf{F}_{EE}$ and A_M.

Sketches giving examples for all SC are depicted in Fig. 5.3.

In robotics, typically used sensors measure not only forces but also torques. Nevertheless, torques are not used for calculating the SC. This is caused by the fact that torque information is not unequivocally because it depends on the force application point and the lever arm. Both are unknown for the considered errors. Thus, a torque cannot be clearly assigned to a specific contact situation.



(a) SC_{FA} (b) SC_{AG}



(c) SC_{AR} (d) SC_D

Figure 5.3.: Sketches illustrating the SC. The SCs are drawn in red.

5.3.1. One Selection Criterion

The simplest approach is to select a strategy from the system's database by considering only one SC. Here, a strategy is chosen for minimizing the differences between the SC of a current error and the SC of the strategies inside the database. So, only one SC is considered and the others are ignored. Therefore, one SC is preferred and it is called SC_{Pref} (cf. Sec. 4.1.3). SC_{Pref} is assembly task dependent and was defined by a domain expert at design time. So, a-priori knowledge regarding error classes within the *HD* is reflected. The disadvantage of using only SC_{Pref} is that not all information available w.r.t. the error is used. Compared to an approach considering all information (cf. Sec. 5.3.2), a faster error handling can be achieved. Another reason for using this simple approach is the low computational cost especially for large databases.

5.3.2. Multi-criteria Optimization

Fixing the disadvantage that not all information available is used, a multi-criteria optimization is carried out. Again, it is the goal to find a strategy s that minimizes $SC(s)$ distances w.r.t. a set $SC_{\{FA,AG,AR,D\},C}$. Here, the index $_C$ is used for referring to the selection criteria of the current error. In order to involve all SC in the selection process, the Pareto front P is calculated according to Eq. 5.1 by optimizing over all available strategies. This approach accepts only improvements and rejects deterioration.

$$P = \min_{\forall s \in DB} \Big\{ SC_{FA}(s) - SC_{FA,C},\ SC_{AG}(s) - SC_{AG,C}, \\ SC_{AR}(s) - SC_{AR,C},\ SC_{D}(s) - SC_{D,C} \Big\} \tag{5.1}$$

The next step is to select a strategy out of P. Thereby, P is the set of Pareto-optimal strategies. Again, this selection is done by utilizing SC_{Pref} which is given by the *HD* (cf. Sec. 4.1.3). By performing the optimization step, other criteria are taken into account. Even if the final selection is made regarding SC_{Pref}.

5.4. Performance Criteria

For each demonstration describing the recovery from an error state it is possible to calculate a performance value by using a performance criteria (PC). Thereby, PC are task

specific and depend also on the error state. Therefore, a matching PC needs to be defined by the domain expert. Thus, one of the error state attributes, defined within the *HD*, is a preferred performance criterion (PC_{Pref}, cf. Sec. 4.1.3).

Generally applicable performance criteria (PC) are for e.g.:

- PC_P: Cartesian path $\sum_{i=1}^{n} \sqrt{\Delta x_i^2 + \Delta y_i^2 + \Delta z_i^2}$ travelled by the end-effector.
- PC_F: Sum of the acting forces $\sum_{i=1}^{n} (F_{i,x} + F_{i,y} + F_{i,z})$.
- PC_T: Time to completion.
- PC_J: Jerk of e.g. a force or velocity profile.

The definition of highly task specific PC is also possible.

5.5. Strategy Fusion

Human demonstrations used in recovery are rarely optimal. Therefore, an approach for optimizing such demonstrations is required and presented here. An optimization is feasible with respect to a preferred performance criteria (PC_{Pref}, cf. Sec. 5.4). In addition, it is only possible to use demonstrations belonging to the same error class for performing an optimization. It is the goal to minimize PC because they correlate to the effort needed to execute an assembly task.

Assembly process characteristics and the limited observability (cf. Sec. 5.1.1) results in restrictions for possible optimizations of human demonstrations. At a microscopic level, the exact state and cause of an error is unknown and it is impossible to determine which parts of a recovery strategy are really needed for recovering. So parts cannot be removed without risking that the strategy stops working. It is also unknown if a human stopped a strategy while sampling the demonstration because the strategy did not work and applied a new one (cf. Sec. 5.1.2).

However, it is possible to stop the execution of a strategy and to apply a different one as long as it is ensured that at least the last strategy is completed. Within this approach, it is not intended that the first strategy results already in a successful recovery. Instead, the first one is only needed to change the SC so that the second strategy can be applied. Now, if the performance according to the chosen PC is better for the executed part of the first strategy and the entire second are better, than the combination of both achieves

an improvement regarding the performance. This approach is called a fusion of strategies and can be applied offline and performed in advance. All strategies stored in a database can be used as an input for the fusion approach. This also includes a strategy derived from a demonstration that has just been performed (cf. Sec. 1.3).

Fusion of Two Strategies

The strategies s_1 and s_2 are exemplary used for explaining how the fusion of strategies works. This example is also depicted in Fig. 5.4.

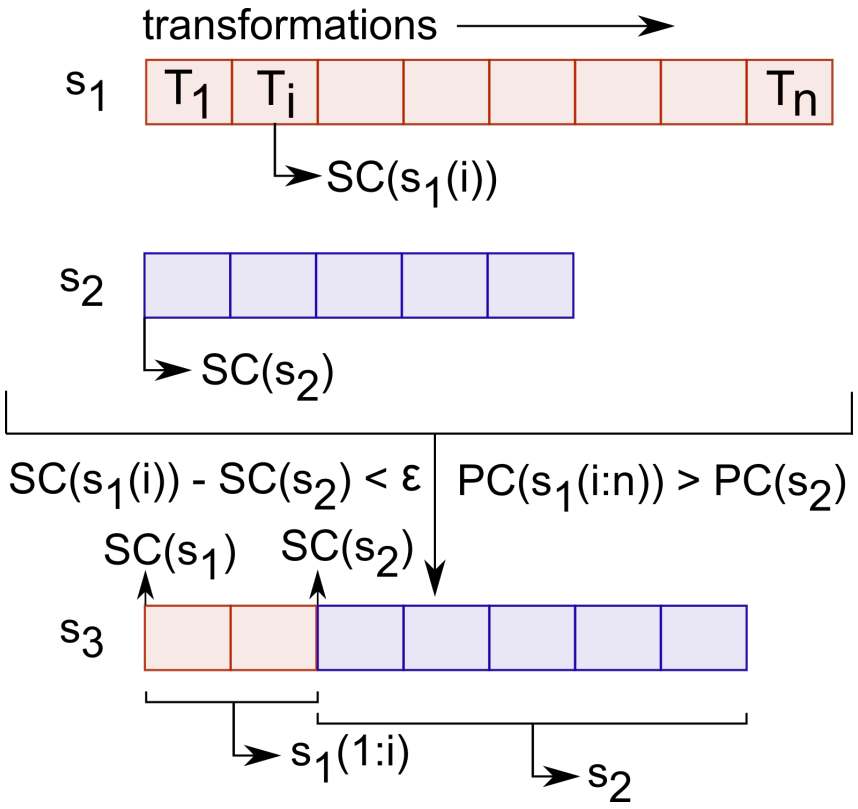


Figure 5.4.: Here, it is shown how to fuse the two strategies s_1 and s_2. The result is s_3 which can be used to recover from the same error as s_1. But s_3 has a better performance than s_1. The fusion is done by finding a step where the SC are similar and the performance score PC for the rest of s_1 is worse than that for s_2. At this point, the rest of s_1 is replaced by s_2.

The strategy s_1 contains $n, n \in \mathbb{N}$ steps or transformations T. s_1 is traversed from step $i = 1, i \in \mathbb{N}$ to $i = n$. The special feature is that the SC are not just used to describe

the starting conditions of a strategy e.g. $SC(s_1)$. Instead, they are also used to describe the current situation at a specific step i e.g. $SC(s_1(i))$. In order to find a spot to do a transition from s_1 to s_2, it is tested if $SC(s_1(i)) - SC(s_2) < \epsilon$ holds. The threshold ϵ, from which it is assumed that the SC are similar, depends on the assembly task. In addition, since it is the goal to improve the performance, $PC(s_1(i:n)) > PC(s_2)$ must be satisfied. Subsequently, both strategies can be fused together. Therefore, s_2 is appended to the beginning of the first $s_1(1:i)$. The result is s_3. This new strategy can be used for the same SC as s_1 but it has a better performance. It must be ensured that now the reference frame for s_2 is F_i of s_1. So discontinuities regarding the orientations of s_3 can be avoided.

Fusion of Multiple Strategies

When applying the fusion approach to multiple strategies inside a database, the strategies are considered in turns. Per iteration, each of the strategies is selected once and it is tried to fuse it with all other available strategies. Subsequently, the fusion result with the best PC is used for replacing the current strategy in the database. It is replaced because otherwise there would be two strategies for exactly the same SC.

One iteration is also called a level $L_j, j \in \mathbb{N}$. The index of a level limits also the maximal number of strategies that can be found in the fusion result. From a user's perspective, it is possible to either limit L_j or to go on until there is one iteration without any fusion. Thus, the computational effort of the fusion approach can be adapted to the task specific requirements.

Chapter 6

Experimental Verification of the Recovery Strategy Approach

This chapter presents an experiment and discusses its findings in order to clarify whether the *HD* (cf. Chap. 4) and the strategy presentation (cf. Chap. 5) provide an approach for recovering from assembly errors based on human demonstrations. Parts of the results have already been published [61, 62].

The objectives of this experiment are:

- Is an error handling based on human demonstrations working and does it offer any added value?
- Is an SC based strategy selection better than a random based one?
- Is the applicability of a special SC error dependent?
- How does the number of successful recoveries change if the Pareto-optimized approach is used instead of only one SC?
- Is there an improvement regarding the performance of a recovery if the fusion approach is applied?
- Does the fusion approach increase the recovery success rate?
- Does the performance of fusion approach change if different input devices have been used for strategy demonstration?

Looking at *HRI*, small benefits for *KG* with stiffness adaptation compared to *KG* without adaptation were found. Nevertheless, for the same reason as already presented in Chap. 3, *KG* is used without adaptation.

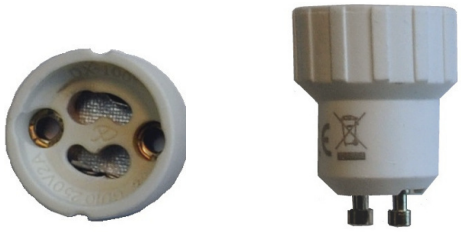
6.1. Assembly Tasks and Errors

During two user studies (cf. Chap. 2 and cf. Chap. 3) data on recovery attempts has already been recorded. Therefore, it is possible to extract strategies and to reuse them in the experiment. As a result, the experimental effort can be reduced. However, there have been mechanical problems with the workpieces of *DIN Rail* and *Bracket*. Thus, these two assembly tasks need to be excluded such that following tasks are remaining:

- *Peg*
- *Spline Shaft*
- *Pleuel - Top*
- *Pleuel - Bottom*

Nevertheless, for covering all types of mating as specified by *DIN 8593-1:2003-09* and visualized in Fig. 1.3, a new assembly task with a spring contact is needed. For this purpose, the new task *GU10* is introduced. This task consists of a GU10 light bulb connector and a corresponding socket with a spring element inside. Both workpieces are depicted in Fig. 6.1.

Engineering drawing for all workpieces used in this experiment are given in Appendix A.



(a) Socket (b) GU10 connector

Figure 6.1.: Workpieces of the *GU10* assembly task.

6.1.1. Characteristics of Assembly Errors

There are differences regarding the error characteristics. This is caused by the circumstance whether a specific error has been reproduced or all errors occurring during an assembly operation have been considered.

Various errors are considered for the assembly tasks *Spline Shaft* and *Peg*, because when carrying out the previous user study (cf. Chap. 3), errors occurred over the entire length of the components during assembly. The characteristic of an error depends on the location it occurs at. Thus, the tasks are subdivided into intervals. For each interval the most frequent errors are chosen. Therefore, 5 and 4 error have been selected for *Spline Shaft* and *Peg*, respectively. The reason is that the socket for the task *Peg* is geometrically shorter compared to *Spline Shaft*. The exact subdivision into intervals as well as the SC describing the errors is presented in Sec. 6.1.2.

In contrast, the assembly task *GU10* was considered anew. Therefore, only one error is specified. This error is given by the following SC:

- $SC_{FA} = 40.35°$
- $SC_{AG} = 45.87°$
- $SC_{AR} = 34.66°$
- $SC_{D} = 8.32\ mm$

This error was chosen because the spring-loaded element is in a contact state. For creating a task specific recover strategy database, only one shop-floor worker demonstrated recoveries to a robot. The reason for this is to show that the presented error handling approach (cf. Sec. 1.3) also works if only strategies demonstrated by one human are used. Specifically, it should be examined whether strategy fusion (cf. Sec. 5.5) can increase the overall system performance. Also, the influence of using only strategies demonstrated by the same human or by different humans is analysed.

For the assembly tasks *Pleuel - Top* and *Pleuel - Bottom*, the errors defined in Sec. 2.1.2 are used. The special feature of the *Pleuel* task is that strategies which were recorded using the various input devices (cf. Sec. 2.2) are used. Therefore, it should be considered whether the presented error handling approach works independently of the input devices.

6.1.2. Subdivision of *Peg* and *Spline Shaft*

The recovery strategies related to *Peg* and *Spline Shaft* have been sampled during a user study (cf. Chap. 3). Thus, no special errors were commanded, but all naturally occurring ones were detected. As a result, the databases of those assembly tasks contain strategies matching lots of different error. In contrast to that, it is crucial for the following experiments (cf. Sec. 6.4) that the same error occurs in every attempt.

That is why the errors are subdivided into intervals. For each interval the statistically most frequently occurring error situation is used for the experiments. This subdivision is made base on the workpiece's z-position at the moment an error occurs. The reason for this is that here the z-position is used to describe the assembly progress. It turns out that in general the error characteristic as well as the respective occurrence probability both depend on the z-position. One possible reason for this may be the increasing mechanical guidance w.r.t. the assembly progress.

Peg

The error distribution over the z-position is shown as a box plot in Fig. 6.2. The subdivision of *Peg* into four intervals, as shown in Tab. 6.1, was set w.r.t. the quartiles, the median and the whiskers of the box plot. For each interval, the average error is used. Its *SC*s are listed in Tab. 6.2.

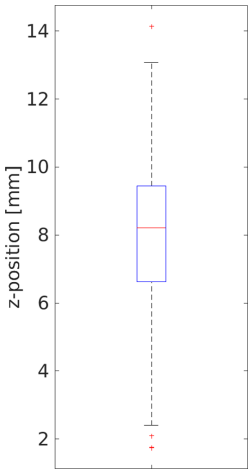


Figure 6.2.: *Peg:* Distribution of errors over the z-position.

Table 6.1.: *Peg:* Subdivision of errors based on the z-position.

Interval	Start	End
1	9.44 mm	13.08 mm
2	8.20 mm	9.44 mm
3	6.61 mm	8.20 mm
4	2.39 mm	6.61 mm

Table 6.2.: *Peg: SC*s w.r.t. the subdivision into intervals.

SC	Statistical Measure	Interval 1	2	3	4
SC_{FA} [°]	Median	133.26	131.62	131.62	132.82
	SD	1.46	1.36	1.49	1.29
SC_{AG} [°]	Median	2.10	2.11	2.11	2.10
	SD	1.77	2.05	1.87	
SC_{AR} [°]	Median	2.61	2.70	2.65	2.61
	SD	1.41	1.35	1.39	1.45
SC_D $[mm]$	Median	0.23	0.23	0.23	0.23
	SD	0.28	0.28	0.28	0.28

Spline Shaft

Regarding the *Spline Shaft* assembly task, the procedure is similar to that of *Peg*. The error distribution over the z-position is shown as a box plot in Fig. 6.3. The subdivision into five intervals, as shown in Tab. 6.3, was set w.r.t. the quartiles, the median and the whiskers of the box plot. For each interval, the average error is used. Its *SC* are listed in Tab. 6.4.

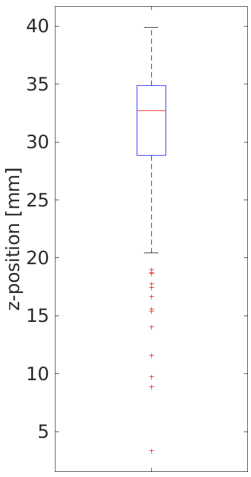


Figure 6.3.: *Spline Shaft:* Distribution of errors over the z-position.

Table 6.3.: *Spline Shaft:* Subdivision of errors based on the z-position.

Interval	Start	End
1	34.87 mm	40.00 mm
2	32.86 mm	34.87 mm
3	30.85 mm	32.86 mm
4	28.84 mm	30.85 mm
5	20.40 mm	28.84 mm

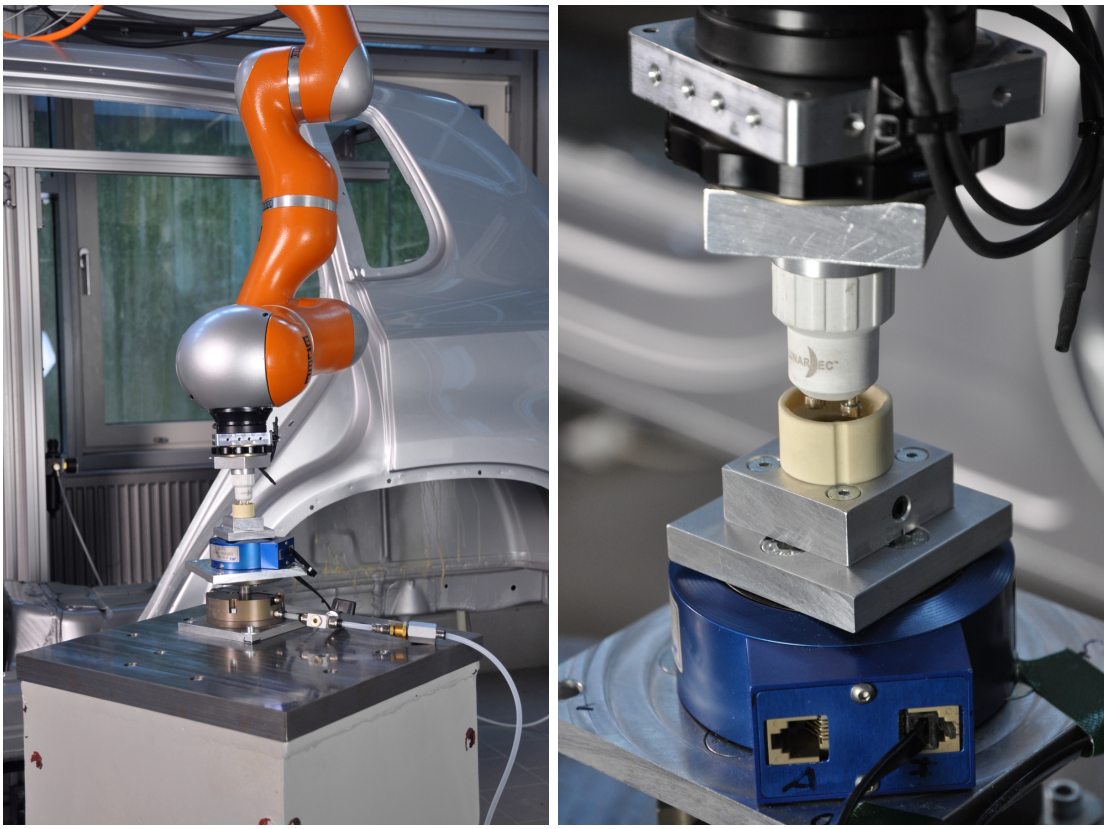
Table 6.4.: *Spline Shaft: SC*s w.r.t. the subdivision into intervals.

SC	Statistical Measure	Interval 1	2	3	4	5
SC_{FA} [°]	Median	145.51	146.12	143.77	147.22	145.51
	SD	1.27	1.33	1.15	1.94	1.49
SC_D [mm]	Median	0.26	0.25	0.25	0.33	0.25
	SD	0.18	0.17	0.18	0.20	0.18
SC_{AG} [°]	Median	1.70	1.69	1.70	2.13	1.70
	SD	1.69	1.70	1.80	2.19	1.73
SC_{AR} [°]	Median	1.46	1.43	1.40	1.78	1.46
	SD	1.58	1.69	1.67	1.63	1.75

6.2. Experimental Setup

Since data from the two previous user studies (cf. Chap. 2 and Chap. 3) is reused in this experiment, the setup is similar to that described in Sec. 2.1 and Sec. 3.2. An overview of the setup is given by Fig. 6.4(a). The most important component is a KUKA Light Weight Robot (LWR IV+). It is mounted upside down, such that the assembly operation can be observed well. Thus, an operator can intervene in an emergency to avoid damage to the workpieces or the hardware.

In every case, one of the workpieces used for an assembly task (cf. Sec. 6.1) was attached to the robot's end-effector. The other workpiece was mounted on a basis platform for carrying out the trials. This platform was located above a force-torque sensor (*JR3 85M35A-140-D*). Although the robot is able to estimate the forces and torques acting on its end-effector, an additional force-torque sensor was used. The reason for this is the very low sensor noise of the *JR3* sensor ($F_{x,y} : +/-0.1$ [N]; $F_z : +/-0.3$ [N]). Also, an overload protection device was used for protecting the sensor from potential damages. For this reason, the device was placed below the force-torque sensor. A detailed view of both workpieces, the basis platform and the force-torque sensor is shown in Fig. 6.4(b).



(a) Overview of the experimental setup.

(b) Detailed view onto the workpieces for *GU10* assembly task, the force-torque sensor and the basis platform.

Figure 6.4.: Robot performing the *GU10* assembly task.

6.3. Experimental Procedure

The experimental procedure differs depending on whether the strategy selection or the fusion approach are investigated. At the beginning of a strategy selection experiment, the robot is commanded to a specific error. Then a recovery strategy is selected. Therefore, different approaches for strategy selection are used. The approaches are random-based selection (cf. Sec. 6.4.1), by using one *SC* (cf. Sec. 6.4.2) and multi-criteria optimization based strategy selection (cf. Sec. 6.4.3). Which *SC* is used in combination with the selection approach, depends on the respective experiment. In all cases, databases containing unmodified strategies have been used. Once a strategy has been selected, it is executed. Although the assembly tasks *Peg* and *Spline Shaft* have been split into intervals, all strategies are still contained in the databases.

If, on the other hand, the fusion approach is considered, strategies are randomly selected. Here, databases optimized by the fusion approach have been used (cf. Sec. 6.4.4). For performing an L_1 fusion, two general applicable PC have been used.

- PC_P: Cartesian path $\sum_{i=1}^{n} \sqrt{\Delta x_i^2 + \Delta y_i^2 + \Delta z_i^2}$ travelled by the end-effector.
- PC_F: Sum of acting forces $\sum_{i=1}^{n} (F_{i,x} + F_{i,y} + F_{i,z})$.

After a strategy has been selected, the robot is commanded to exactly that error specified by the strategy's SC. Subsequently, it is executed.

During strategy execution, the acting forces as well as the robot's motions are measured in order to use this data for the evaluation of the fusion approach. A recovery is conducted as successful if it is possible to continue with the assembly operation after a recovery strategy has been applied. Once a strategy has been carried out, the robot is commanded back to the starting position. The whole process is monitored by an operator to intervene in the event of a fault and to avoid hardware damage.

Although only specific errors are commanded to the robot, different strategies can be selected. The reason is the robot's limited accuracy when it is commanded to a pose. Note that while other approaches have sought ways to minimize these inaccuracies, in this experiment they are desired [71]. Due to the robot's pose inaccuracy, small variations occur regarding the error causes and thus the SC take effect. It should be noted that the specified accuracy (*ISO 9283*) of the robot's end-effector is $+/- 0.05\ mm$ [45].

6.3.1. Strategy Databases

The number of strategies in a database used within an experiment is listed in Tab. 6.5. For each assembly task, a separate database was used so that only strategies belonging to a specific assembly task were used. In general, the number of strategies inside a database depends on the underlying user study (cf. Sec. 6.1.1). However, if the fusion approach is used, the number is also affected by used thresholds (cf. Sec. 6.3.2) and applied PC.

Table 6.5.: This table lists how many strategies are included in a database and therefore used for an experiment. Here, the term *'Pure Strategies'* means that a strategy is used as it has been recorded. So, no processing, e.g. fusion, has been applied.

Assembly Task	Mode	Number
Peg	Pure Strategies	377
	Fusion - PC_P	47
	Fusion - PC_F	131
Spline Shaft	Pure Strategies	389
	Fusion - PC_P	99
	Fusion - PC_F	214
Pleuel - Top	Pure Strategies	309
	Fusion - PC_P	286
	Fusion - PC_F	280
Pleuel - Bottom	Pure Strategies	309
	Fusion - PC_P	291
	Fusion - PC_F	292
GU10	Pure Strategies	150
	Fusion - PC_P	11
	Fusion - PC_F	13

6.3.2. Thresholds and Statistics of the Fusion Approach

The fusion approach requires that SC are compared. Since the SC are floating-point numbers and there is some process noise in the calculation of SC, a threshold must be defined. It defines the maximal numeric difference between SC. If it is below a threshold then SC can be considered as the same. This threshold is specified by a domain expert because it is assembly task dependent. The thresholds used here are given by Tab. 6.6.

Table 6.6.: Thresholds used for comparing SC. These values define that the maximal numeric difference between two SC to be considered equal.

	Peg	*Spline Shaft*	*Pleuel*		*GU10*
			Top	*Bottom*	
ΔSC_{FA} [°]	0.50	0.50	1.00	1.00	1.00
ΔSC_D [mm]	0.25	0.25	0.50	0.50	0.50
ΔSC_{AG} [°]	0.50	0.50	0.50	0.50	1.00
ΔSC_{AR} [°]	0.25	0.25	0.50	0.50	0.50

One use case of these thresholds is to determined how often a strategy is found within others. This is necessary for being able to use the fusion approach (cf. Sec. 5.5). Therefore, the SC at the beginning of a strategy need to occur within other strategies. So, it is feasible to fuse them together. An overview how often this was possible is given by Tab. 6.7. However, it needs to be kept in mind that a fusion is only performed if the overall performance gets improved. Nevertheless, the amount of strategies inside the databases, as presented in Tab. 6.5, is caused by the applied thresholds.

Table 6.7.: Mean and standard deviation of how often the SC of a strategy occur in all other strategies.

	Peg	*Spline Shaft*	*Pleuel*		*GU10*
			Top	*Bottom*	
$Mean(Number\ of\ Occurrence)$	2.32	5.33	127.97	115.47	30.49
$SD(Number\ of\ Occurrence)$	2.43	9.01	103.58	83.03	26.45

6.4. Experimental Results

All experiments were conducted in contact situations. Therefore, unsupervised experiments were not feasible. The robot needed be monitored so that an emergency shutdown can be done in case of a failure. It was also necessary to examine the error reproduction for obtaining meaningful results. Additionally, in case of system failures, e.g. triggering of the overload protection device or too high joint torques causing a robot emergency stop, the experiment needed to be restarted. Therefore, the experiments were very time-consuming. For this reason, only 25 repetitions were performed per specific trial. Nevertheless, 4440 trials have been carried out overall and the results are presented here.

6.4.1. Random-based Strategy Selection

The simplest strategy selecting approach is to use random. But no information regarding an error is used in this case. Thus, the results can be used as a reference for evaluating other strategy selection approaches. These results are given by Tab. 6.8 column SC_R. The effectiveness of the multi-criteria optimization (cf. Sec. 5.3.2) is assessed by first determining an optimal set and then making a random selection w.r.t. this set. The results are shown in Tab. 6.8 column SC_{PR}. For each cell of this table, 25 trials were performed. So, a total of 600 recovery attempts were carried out.

Table 6.8.: Random-based selection of a strategy regarding all strategies or an optimized set. For each cell of this table **n=25** trials have been carried out. The table is showing the number of successful recoveries.

		SC_R	SC_{PR}
Peg	Interval 1	19	21
	Interval 2	18	22
	Interval 3	20	22
	Interval 4	22	23
Spline Shaft	Interval 1	14	18
	Interval 2	15	19
	Interval 3	18	21
	Interval 4	20	21
	Interval 5	20	22
Pleuel	*Top*	18	22
	Bottom	19	22
GU10		17	21

6.4.2. Selection Criterion based Strategy Selection

Limited information regarding an error can be utilised by using one SC. So, the error is described and a matching recovery strategy can be selected (cf. Sec. 5.3.1). The results are given by Tab. 6.9. Here, the number of successful recoveries is shown. For each combination of an assembly task and a SC, 25 trials were performed. Thus, overall 1200 recovery strategies have been executed in order to create that table.

Table 6.9.: Strategy selection is done by using only one SC. For each cell of this table **n=25** trials have been carried out. The table is showing the number of successful recoveries.

		SC_{FA}	SC_D	SC_{AG}	SC_{AR}
Peg	Interval 1	21	20	23	20
	Interval 2	22	21	23	23
	Interval 3	24	21	23	21
	Interval 4	25	24	25	23
Spline Shaft	Interval 1	21	17	19	17
	Interval 2	20	15	22	20
	Interval 3	22	21	21	22
	Interval 4	23	22	23	20
	Interval 5	25	22	24	22
Pleuel	*Top*	25	19	19	21
	Bottom	25	20	25	21
GU10		21	19	22	24

6.4.3. Pareto-optimized Strategy Selection

All available information w.r.t. an error is used by first executing a Pareto-optimization and then selecting a strategy out of the optimized set according to one SC (cf. Sec. 5.3.2). For each combination of an assembly task and a SC, 25 trials were performed. So, the results, which are presented in Tab. 6.10, are based 1200 carried out trials.

Table 6.10.: Strategy selection based on Pareto-optimization and usage of SC_{Pref}. For each cell of this table **n=25** trials have been carried out. The table is showing the number of successful recoveries.

		SC_{FA}	SC_D	SC_{AG}	SC_{AR}
Peg	Interval 1	25	21	24	23
	Interval 2	24	23	24	23
	Interval 3	25	23	25	23
	Interval 4	25	24	25	24
Spline Shaft	Interval 1	22	21	21	21
	Interval 2	22	18	24	23
	Interval 3	24	23	23	22
	Interval 4	24	23	23	22
	Interval 5	25	23	25	24
Pleuel	*Top*	25	22	23	22
	Bottom	25	24	23	25
GU10		23	22	23	25

6.4.4. Performance Gains by the Fusion Approach

For *Peg* and *Spline Shaft* the separation into different intervals was dropped. The reason is that now the focus is on increasing the performance w.r.t. to given PC instead of considering only the number of successful recoveries. Nevertheless, the *Pleuel* is treated separately, since *Top* and *Bottom* are different tasks.

Here, two generally applicable PC are used as already mentioned in Sec. 6.3. For determining the percentage improvement, the experiments carried out here were compared with the demonstrations. The improvement w.r.t. to both PC is given by Tab. 6.11. The number of successful recoveries and the amount of performed trials are also given. In total, 1440 trials have been carried out.

Table 6.11.: Fusion of two strategies for increasing the assembly operation performance. Therefore, two *PC* are used so that different aspects of the assembly get improved.

		Peg	*Spline Shaft*	*Pleuel* *Top*	*Bottom*	*GU10*
PC_P	*Trials*	250	250	60	60	100
	Number of successes	248	245	59	58	98
	Improvement	18.43 %	19.02 %	28.43 %	17.11 %	18.99 %
PC_F	*Trials*	250	250	60	60	100
	Number of successes	247	249	55	59	99
	Improvement	32.12 %	30.51 %	34.55 %	25.64 %	38.10 %

It should be noted that different input devices (cf. Sec. 2.2) have been used while demonstrating the recovery strategies for *Pleuel*. With regard to the fused strategies, it can be said that all input devices are equally represented.

6.5. Discussion

The results shown in Tab. 6.8, 6.9, 6.10 and 6.11 are based on a total of 4400 experimental trials. In general, these have been successful recoveries. Therefore, it is possible to conclude that the proposed error handling approach (cf. Sec. 1.3) and especially the introduced strategy representation (cf. Chap. 5) is working as good as expected.

Looking at the results for SC_R in Tab. 6.8 it can be concluded that a random strategy selection is working but does not succeed in all trials. Nevertheless, all used recovery strategies have been demonstrated for the same error state of the decomposed assembly task. So, there must be more precise differences regarding these errors which are not covered by the *HD*. Otherwise, randomly selected strategies would always lead to successful recoveries. Thus, it is possible to use the values obtained for SC_R as the ground truth for the evaluation of other selection approaches. Checking the mentioned results, it can be shown that the usage of a *SC* and especially the Pareto-optimization approach have a positive influence. The column SC_{PR} contains the number of successful recoveries for first calculating an optimal set by using all *SC* and then performing a random selection w.r.t. this optimal set. Here, it can be noticed that the number of successful recoveries is increased for all assembly tasks compared to the results for SC_R. This fact emphasizes that using *SC* can overcome a rough *HD*. Using all available information by performing the Pareto-optimization approach is a commendable way for strategy selection.

For determining the influence and characteristics of SC, only one SC is used for strategy selection. The results are depicted in Tab. 6.9. By looking at that table, it can be seen that the number of successful recoveries is in most cases higher than using a random strategy selection. Only for *Spline Shaft* (Interval 2) and SC_D the achieved result is similar. Therefore, it can be concluded that using a SC is better than using random. If one SC is used then it is the SC_{Pref} encoded as state attribute in the *HD*. So, the domain expert needs to be careful while defining SC_{Pref}. This is caused by the fact that SC are assembly task dependent. Applying an improper SC can unnecessarily reduce the number of successful recoveries. This is shown by the fact that SC_{AG} is two times the best SC for *Peg*. On the other hand, SC_{FA} is the best one for *Spline Shaft*. For *Pleuel*, the picture is more differentiated. While SC_{FA} is the best for *Top*, similar results are also achieved by SC_{AG} for *Bottom*. At *GU10*, it looks completely different. Here, the best result is achieved using SC_{AR}.

The highest numbers of successful recoveries has been achieved by using the Pareto-optimization approach. These results are given by Tab. 6.10. In general, the mean of successful recoveries has increased from 21.65 for Tab. 6.9 to now 23.25. Only for *Pleuel - Bottom* the results got worse. But on the other hand, 11 times all 24 performed recoveries have been successful. This has been only 6 times the case for using one SC. At the same time, the standard deviation decreased from 2.26 for Tab. 6.9 to now 1.44. Therefore, it can be concluded that the Pareto-optimization approach is better than only using one SC. So, this approach should be preferred for strategy selection. Another point is that the impact of a wrong SC_{Pref} is reduced. This result is caused by the smaller standard deviation of the optimization approach. Nevertheless, SC_{Pref} is still assembly task dependent.

In Tab. 6.11, the experimental results of the fusion approach evaluation can be found. Here, it is obvious that the overall system performance has been increased in general. However, the amount of improvement depends on the assembly task and the considered PC. Regarding PC_P the values for *Peg*, *Spline Shaft* and *GU10* are close. Only those values of both *Pleuel* tasks differ significantly. An even stronger task dependency can be observed for PC_F. The improvement also depends on the considered PC. This is confirmed by comparing the mean improvements w.r.t. the PC. The mean for PC_P is 20.40 % and that for PC_F is 32.11 %.

A special point is that the fusion approach is capable of processing strategies demonstrated by using different input devices. This is the case for the strategies related to *Pleuel* (cf. Sec. 6.4.4). Here, it was also possible to run the fusion approach and to get optimized strategies (cf. Tab. 6.5 and Tab. 6.7). The overall system performance was improved. It is noticeable that regarding PC_P, the highest and also the lowest improvement compared to the other assembly task has been achieved. A similar behaviour can also be found regarding PC_F. So, it can be assumed that this behaviour is more related to the assembly task instead of the used input devices. Here, it can be said, that the fusion approach

is working independently of the used input device and can be applied even if different devices are used for demonstrating recovery strategies.

Another point is, that the fusion approach is also working fine if strategies demonstrated by a single human are fused together. This has been the case for *GU10*. No significant differences compared to other assembly tasks can be found. Therefore, it is possible to say, that the error handling approach (cf. Sec. 1.3) can be used by a single human or a group.

Since the result of a strategy fusion contains the elements of two strategies, a slightly improved recovery rate was expected. However, the rates are not significantly different from those presented in Tab. 6.10. A possible reason could be that only an L_1 fusion was performed and maybe this effect becomes visible only at higher fusion levels. But, by comparing the recovery rates of the fusion approach with them of Tab. 6.10 it can be said that fusion of strategies does not decrease the recovery rate. Therefore, the fusion approach only influences the system performance and not the recovery rate.

Chapter 7

Conclusion

Based on a scenario where humans and robots share a work space and are free to interact, a new approach for recovering robots from error during assembly operations has been presented. Here, recovery strategies are derived from human demonstrations so that they encode the human task-specific knowledge and experience. The presented approach transfers human strategies and knowledge to robots. The success was shown in the experimental section by considering typical assembly errors. In the experiments, a robotic error recovery was made possible by the imitation of a human demonstration. In addition, it was shown how the suboptimal human demonstrations can be optimized automatically.

The following points needed to be addressed for realizing the presented error handling approach:

- Input Device for *HRI* w.r.t. assembly scenarios
- Characteristics of *KG*
- Assembly task representation
- Strategy representation
- Strategy optimization

A detailed summary of the scientific results is given for each of these points in a dedicated paragraph.

Input Device for *HRI* w.r.t. Assembly Scenarios A user study was carried out to answer the question which input device is best suited for *HRI* in an assembly scenario. Only input devices which can be found in typical robot work cells have been considered.

Results from the user study show that *KG* is the best method for interaction in industrial assembly scenarios in terms of both performance and user satisfaction. The reason can be attributed to the fact that a user gets direct tactile feedback during the interaction which makes the *KG* methods more intuitive and less difficult. This can be verified by the results of the user study.

KG shows good performance regarding different criteria, e.g. it is at least 10 times better when analysing the interaction duration. In addition to this, analysis of learning curves shows that *KG* is the most intuitive way for human interaction with the robots. Even a relatively small sample of trials resulted in satisfactory interaction and superior performance. Hence, *KG* can be regarded as suitable for untrained and inexperienced users.

Characteristics of *KG* A second user study was carried out to determine the characteristics of *KG*. It was shown that substantial evidence for rapid learning effects could be found through confirming the ease of learning attributed to *KG*. Surprisingly, neither a general correlation between the performance in manual assembly and that found in *KG* nor a clear dependency on personal attributes could be observed.

A learning effect does exist, especially regarding the first and the second task execution. If *KG* is used, the effect is larger than when solely considering manual assembly. More complex assembly tasks could require more repetitions. A learning effect also occurs when executing the same assembly task but utilising different methods. Participants who assemble manually before using *KG* perform better in their first *KG* trial on the same task. However, this performance gain decreases gradually and is no longer observed in the final trial. After several repetitions of a specific task, participants achieve similar performance. This observation indicates that *KG* is a suitable method for programming robots by shop-floor workers. Thus, the suitability of this input method for rapid learning is again confirmed.

Solving an assembly task manually overwhelmingly results often in the best performance regarding required contact duration and applied forces. The relative complexity variations between tasks observed in manual assembly generally cannot be transferred to *KG*. There was no conclusive evidence that the performance regarding an assembly task depends on personal attributes in general. A slight dependence on spatial sense is the only attribute dependency that could be found. Participants who show good performance in manual assembly do not necessarily perform well when using *KG*. No general statistical evidence for a correlation between the performance regarding *KG* and manual assembly could be

observed.

Furthermore, this second user study suggests that human strategies cannot be extracted easily via *KG*. Comparing manual assembly with *KG*, substantial differences in required contact duration, applied forces, and movement frequencies can be observed. Thus, deriving human strategies by *KG* is likely to yield significantly distorted results. Extracting and transferring strategies to robots using *KG* is possible, but the effects caused by reduced tactile information, inertia and friction losses of the robot have to be considered.

Assembly Task Representation The Hierarchical Decomposition (*HD*) was introduced as a formal and abstract representation of assembly tasks. The *HD* is not solely limited to the assembly domain and *KG*. It can also be used as a formal representation for any kind of task execution, if the required input data is captured during task execution.

Each state and condition of the *HD* are human-understandable because they are based on the task specific knowledge and experience of shop-floor workers and domain experts. Structure and depth of the decomposition can vary over the described assembly operation. This is due to the circumstance that various segments of this process have different levels of priority from a domain expert's view. Usage of the *HD* enables error detection, classification and also prediction.

A speciality is that state attributes are defined for each error state in the *HD*. So, the detailed description of an assembly error and the selection of a matching recovery strategy are possible. These state attributes contain all necessary information regarding the error characteristics.

For these reasons, the *HD* provides the basis to detect and classify errors but also to be able to select and apply the appropriate recovery strategy.

Strategy Representation The presented strategy representation is characterized by the invariance against rotational and translational displacement. Therefore, the re-usability of a strategy is improved and the number of recoveries demonstrated by a human is decreased. Another important point is that strategies are robot independent and can be applied in similar setups w.r.t. the same assembly task. In experiments, the functionality and the applicability of strategy representation were shown.

Several recovery strategies are assigned to the same assembly error. The reason for this is that the *HD* does not dissolve precise enough. But this limitation was successfully solved by the introduction of selection criteria (*SC*). Through the implementation of various *SC*, the strategy matching to an error can be selected. It was shown in experiments that

using a singular SC increases the recovery rate compared to a random-based strategy selection. Another experimental result was the discovery that the influence of selecting a SC is assembly task dependent. However, this debilitating circumstance could be reduced by using a multi-criteria optimization w.r.t. all SC before selecting a strategy. Using this method, the highest possible recovery rate was achieved in the experiment.

Strategy Optimization When optimizing strategies, there is the challenging problem that human intent is unknown. Thus, a strategy cannot be subdivided and there must be a complete strategy as a result of the optimization. This particular challenge is successfully addressed by the presented fusion approach. Here, only parts of a strategy are replaced through the completion of other strategies. An optimization is possible with regard to various performance criteria (PC). In all experiments, using various PC, it was always possible to achieve a clear improvement of the system performance.

A special feature is that the optimization approach is not limited to a specific input device for demonstrating strategies. It is even possible to optimize strategies demonstrated by using different input devices. This optimization has been confirmed by the experiment results. Thus, it is possible for shop-floor workers to use the input device they prefer and yet the strategies can be used and optimized without any problems.

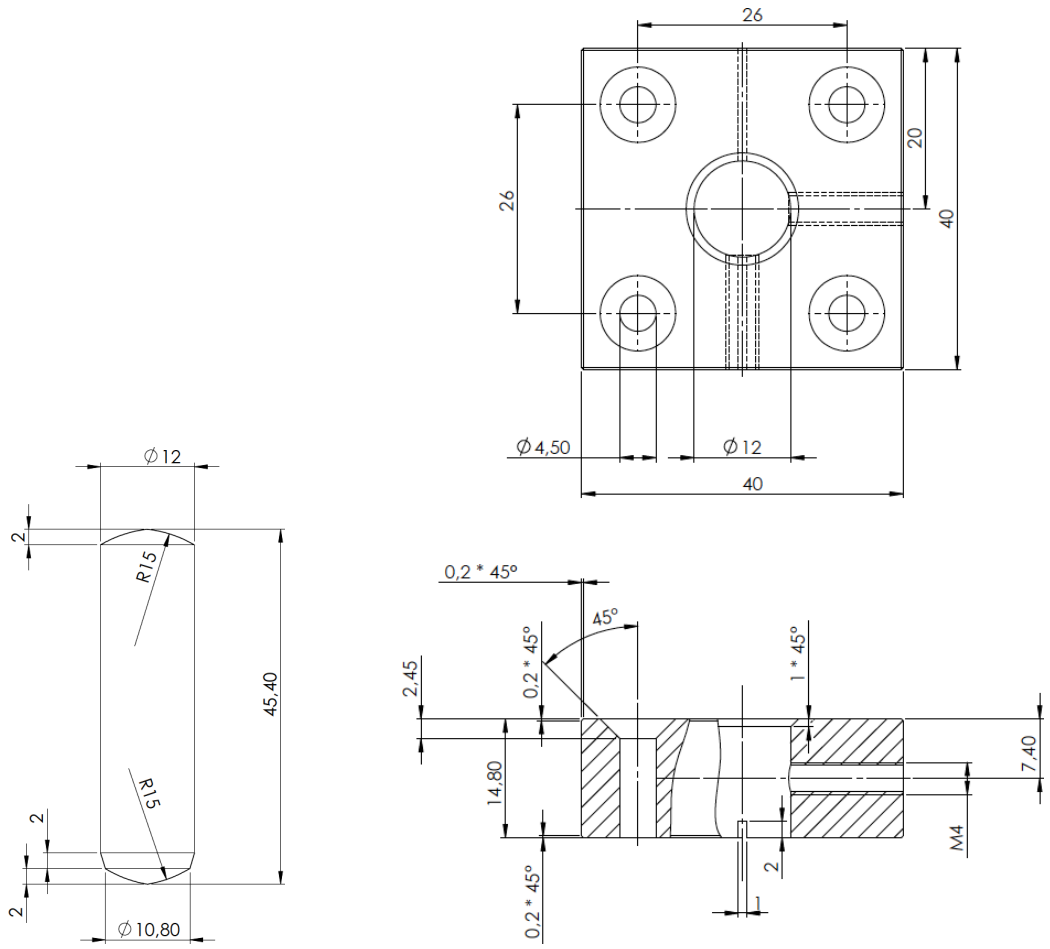
Appendix A

Engineering Drawings

The engineering drawings and additional information regarding the used workpieces for the assembly are presented here. Thus, the performed experiments are comprehensible and reproducible.

A.1. *Peg* Assembly Task

The workpieces for the *Peg* assembly task are custom-made. Here, the tolerance for the peg is $12^{0}_{-0.011}\ mm\ h6$ and $12^{+0.018}_{0}\ mm\ H7$ for the hole. Fig. A.1(a) shows the peg. A plate with the corresponding hole is depicted in Fig. A.1(b).



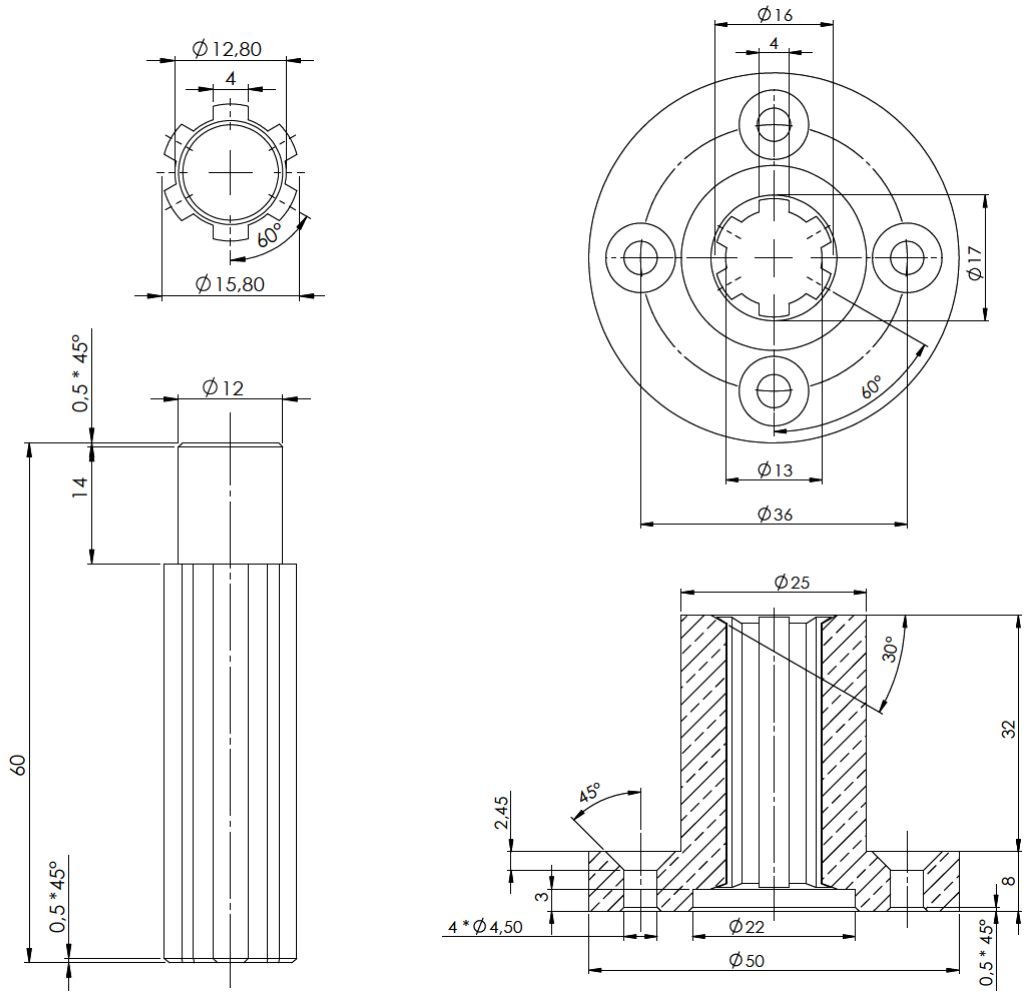
(a) Peg.

(b) Base used for *Peg*.

Figure A.1.: Engineering drawings of a peg and a plate with a hole. Both are used for the assembly task *Peg*.

A.2. *Spline Shaft* Assembly Task

A spline shaft which is depicted in Fig. A.2(a) is used for this assembly task. Also, the corresponding socket is shown in Fig. A.2(b). The tolerances of both workpieces are specified by *DIN 14 KN 13x16*.



(a) Spline shaft.

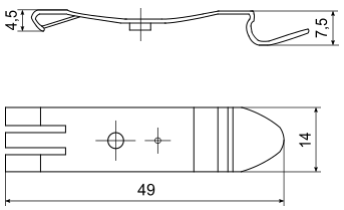
(b) Socket used for *Spline Shaft*.

Figure A.2.: Engineering drawings of a spline shaft and a corresponding socket. Both are used for the assembly task *Spline Shaft*.

A.3. *DIN Rail* Assembly Task

A standardized *DIN* rail (TS 35) and a corresponding *DIN* rail clip are used for the assembly task *DIN Rail*. This clip is produced as model FM 4 by Weidmüller. A DIN rail with a mounted clip is depicted in Fig. A.3(a) and engineering drawings of that clip are given by Fig. A.3(b) .

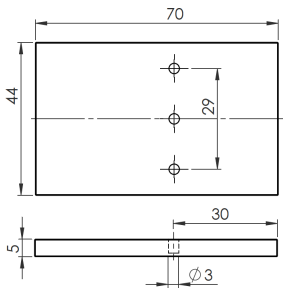
(a) Clip FM 4 mounted on a *DIN* rail TS 35.



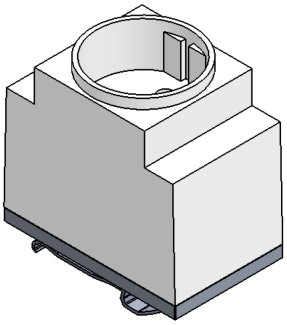
(b) Clip FM 4 as engineering drawing.

Figure A.3.: *DIN* rail and clip.

A realistic industrial assembly task is created by combining the clip with a socket. Therefore, one problem is the mechanical stress of the socket's thin plastic structure. A socket adapter, as shown in Fig. A.4(a), was used to solve this problem. The adapter has three tapped holes. So three clips can be mounted. The overall workpiece is depicted in Fig. A.4(b).



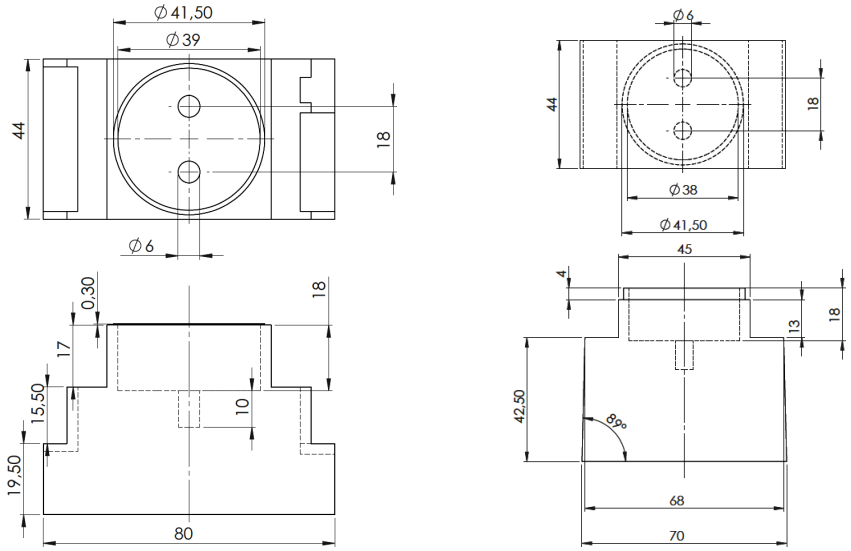
(a) Socket adapter.



(b) CAD model of Socket B with clips and socked adapter.

Figure A.4.: Socket adapter in combination with one socket.

During the user study (cf. Chap, 3) two different sockets (cf. Fig. A.5) had to be used. The reason is that the plastic of *Socket A* was brittle. After 20 participants had performed this assembly task, the socket was broken. Because of this, *Socket B* was used for the rest of the study. Nevertheless, the same clips and the same socket adapter were used for both sockets. Therefore, the contact properties between the clip and the rail was not influenced by changing the sockets. Engineering drawings of both sockets are depicted in Fig. A.5.



(a) Socket A. (b) Socket B.

Figure A.5.: Engineering drawings of both used sockets.

A.4. *Bracket* Assembly Task

The assembly task *Bracket* is difficult. The reason for this is not only the complicated geometry of the workpieces. Also, the pegs' translational *DoF* has an influence.

Fig. A.6 shows a CAD model of the mounting bracket. It consists of four parts which are shown in Fig. A.7. All of this parts are labelled with a number so that they can be identified in Fig. A.6.

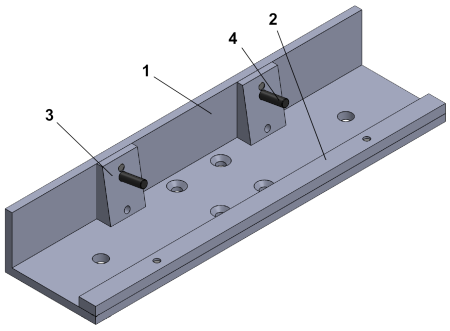
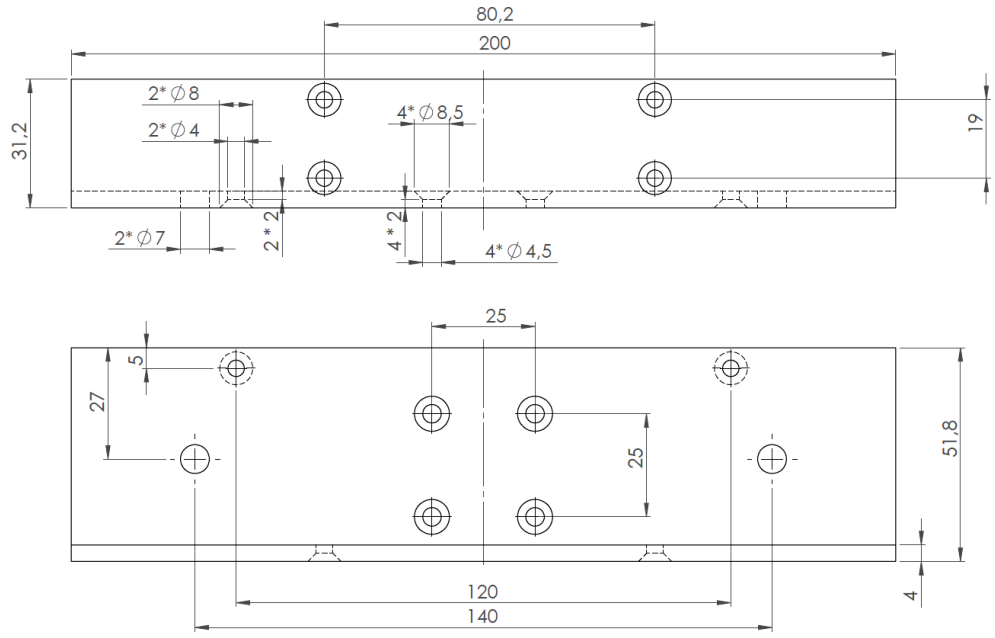
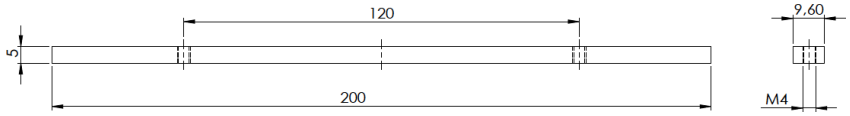


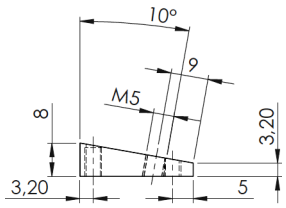
Figure A.6.: CAD model of the mounting bracket's socket.

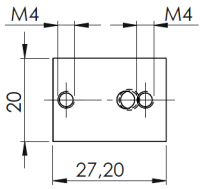


(a) Part one.

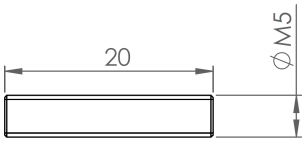


(b) Part two.





(c) Part three.



(d) Part four.

Figure A.7.: Components of the mounting bracket.

A CAD model of the adherend is depicted in Fig. A.8. The adherend is build up from a main component (cf. Fig. A.9), two nuts (cf. Fig. A.10(a)), two pegs (cf. Fig. A.10(b))

and two times two washes (cf. Fig. A.10(c) and cf. Fig. A.10(d)) which differ regarding their geometry.

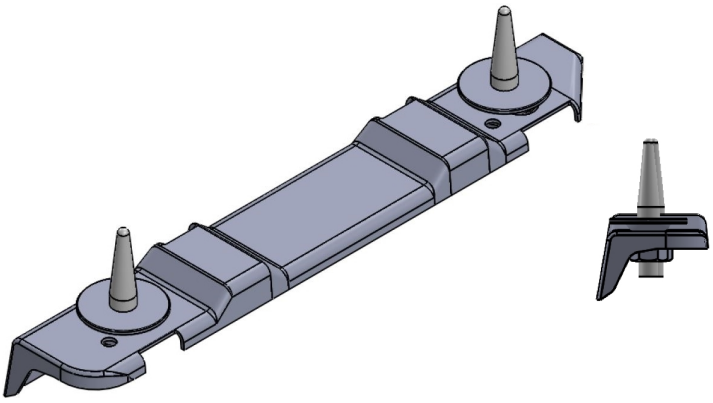


Figure A.8.: CAD model of the adherend.

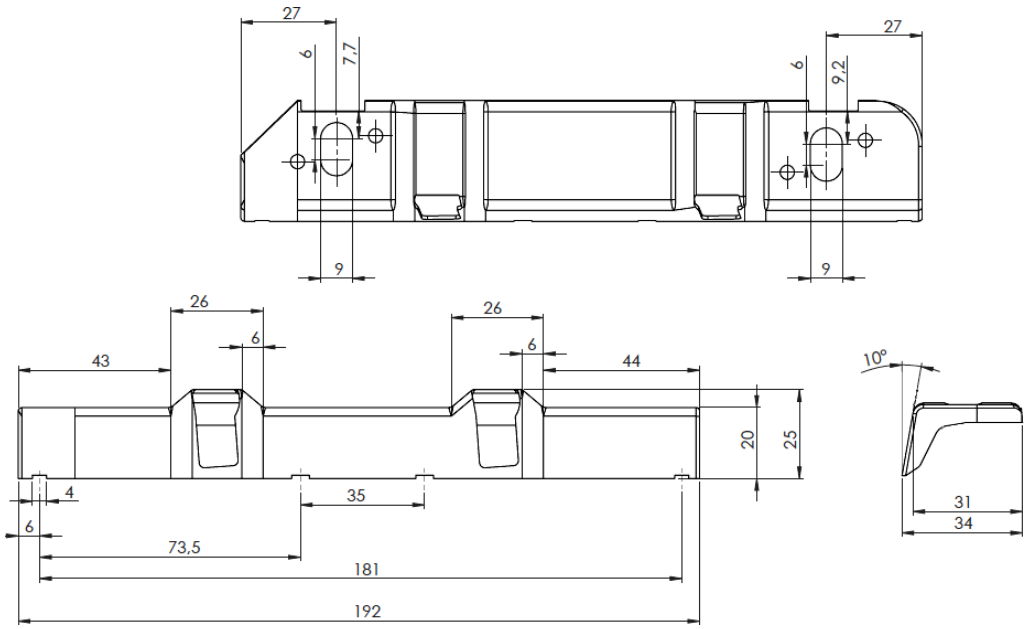
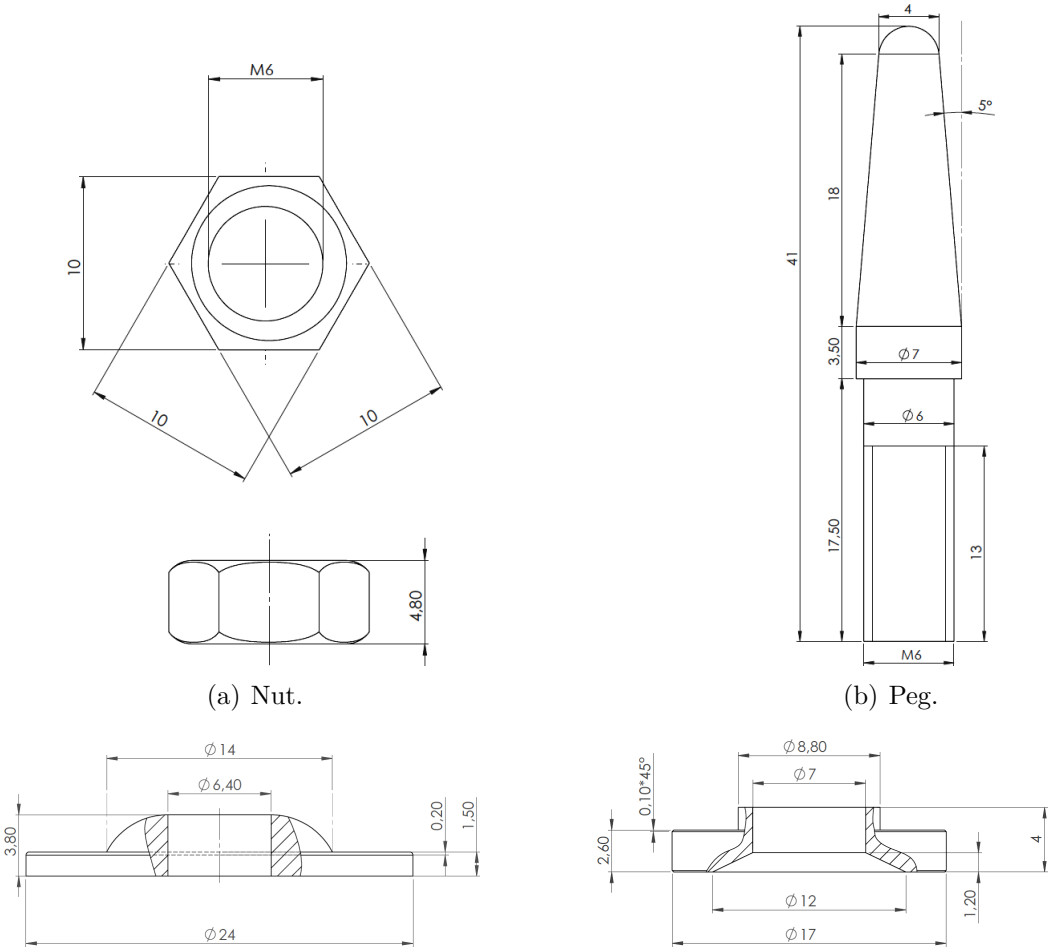


Figure A.9.: Main component of the adherend.



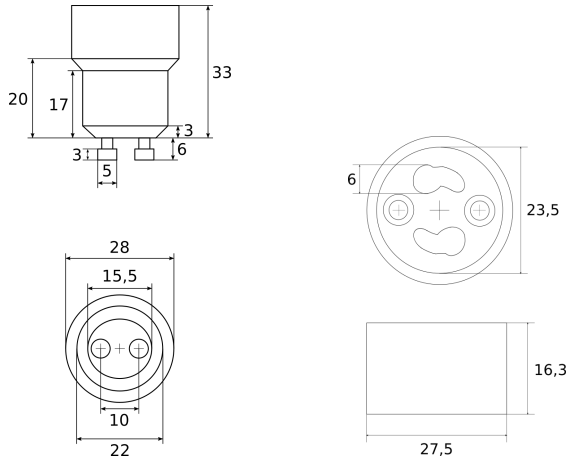
(a) Nut. (b) Peg.

(c) Washer 1. (d) Washer 2.

Figure A.10.: Components of a adherend's assembly group.

A.5. *GU10* Assembly Task

The workpieces which are used for the *GU10* assembly task are depicted in Fig. A.11. Since these workpieces refer to the standard *IEC 60061-1 (7004-121)*, only the main and connection dimensions are given here.



(a) GU10 connector. (b) GU10 socket.

Figure A.11.: Workpieces of the *GU10* assembly task.

A.6. *Pleuel* Assembly Task

A conrod with a notch and a shaft with two keys on it are used for the assembly task *Pleuel*. The conrod is depicted in Fig. A.12 and the shaft in Fig. A.13. For assembly, not only the conrod must be placed on the shaft. Also the notch must be aligned so that it slides over the keys. Therefore, the complexity of this assembly task is specified by the tolerance. Between key and notch there is a clearance of 0.26 *mm*. A higher clearance of 0.44 *mm* exists between the shaft and the conrod.

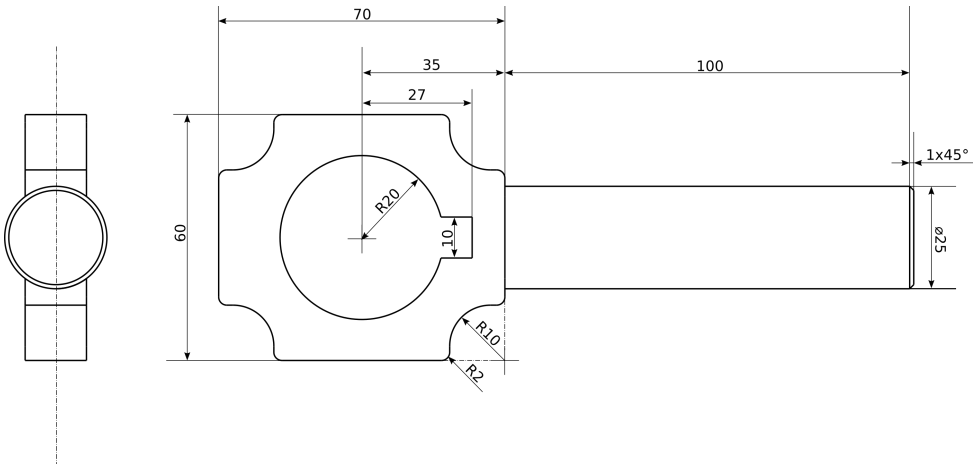


Figure A.12.: A conrod with a notch. It is used as workpiece for the assembly task *Pleuel*.

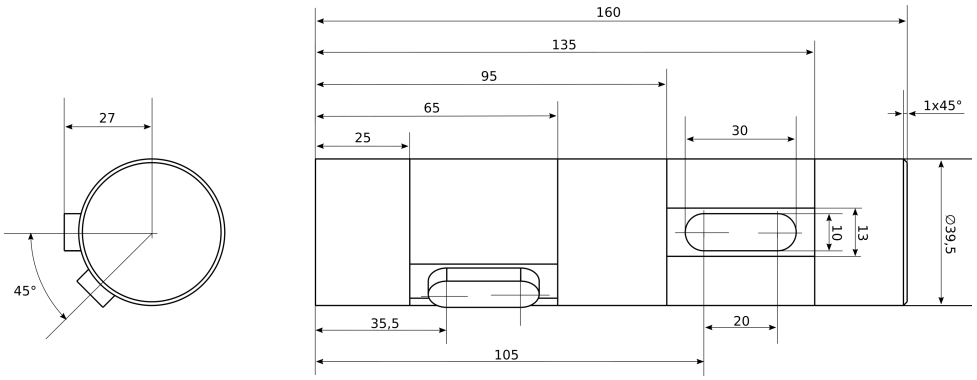


Figure A.13.: Two keys are mounted with an angular offset on a shaft. This workpiece is used for the assembly task *Pleuel*.

List of Abbreviations

DoF	Degree of Freedom
DTCWT	Dual-Tree Complex Wavelet Transform
HD	Hierarchical Decomposition
HMM	Hidden-Markov-Model
HRC	Human-Robot Collaboration
HRI	Human-Robot Interaction
KG	Kinesthetic Guidance
MCP	Manual Control Pendant
PbD	Programming by Demonstration
PCA	Principal Components Analysis
PC	Performance Criteria
SC	Selection Criteria

List of Figures

List of Tables

Bibliography

[1] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[2] B. Akgun, M. Cakmak, J. W. Yoo, and A. L. Thomaz. Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective. In *7. Annual ACM/IEEE International Conference on Human-Robot Interaction*, pages 391–398, 2012.

[3] R. Andre and U. Thomas. Anytime assembly sequence planning. In *47st International Symposium on Robotics*, pages 1–8, 2016.

[4] R. Andre and U. Thomas. Error robust efficient assembly sequence planning with haptic rendering models for rigid and non-rigid assemblies. In *IEEE International Conference on Robotics and Automation*, 2017.

[5] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[6] A. Aryania, B. Daniel, T. Thomessen, and G. Sziebig. New trends in industrial robot controller user interfaces. In *IEEE 3rd International Conference on Cognitive Infocommunications*, pages 365–369, 2012.

[7] T. Asfour, F. Gyarfas, P. Azad, and R. Dillmann. Imitation learning of dual-arm manipulation tasks in humanoid robots. In *6th IEEE-RAS International Conference on Humanoid Robots*, pages 40–47, 2006.

[8] L. Balletti, A. Rocchi, F. Belo, M. Catalano, M. Garabini, G. Grioli, and A. Bicchi. Towards variable impedance assembly: The vsa peg-in-hole. In *12th IEEE-RAS International Conference on Humanoid Robots*, pages 402–408, Nov 2012.

[9] Y. Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, Jan. 2009.

[10] A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Robot programming by demonstration. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 1371–1394. Springer, 2008.

[11] R. Bischoff, J. Kurth, G. Schreiber, R. Koeppe, A. Albu-Schäffer, A. Beyer, O. Eiberger, S. Haddadin, A. Stemmer, G. Grunwald, et al. The kuka-dlr lightweight robot arm-a new reference platform for robotics research and manufacturing. In *41st international symposium on Robotics and 2010 6th German conference on robotics*, pages 1–8. VDE, 2010.

[12] J. Bös, A. Wahrburg, and K. D. Listmann. Iteratively learned and temporally scaled force control with application to robotic assembly in unstructured enviroments. In *International Conference on Robotics and Automation*. IEEE, 2017.

[13] J. F. Broenink and M. L. Tiernego. Peg-in-hole assembly using impedance control with a 6 dof robot. In *Simulation in Industry, 8th European Simulation Symposium*, 1996.

[14] E. Burdet, D. W. Franklin, and T. E. Milner. *Human robotics: neuromechanics and motor control*. MIT Press, 2013.

[15] F. Chen, F. Cannella, J. Huang, H. Sasaki, and T. Fukuda. A study on error recovery search strategies of electronic connector mating for robotic fault-tolerant assembly. *Journal of Intelligent & Robotic Systems*, 81(2):257–271, 2016.

[16] X. W. Chen and S. Y. Nof. *Automating Errors and Conflicts Prognostics and Prevention*, pages 503–525. Springer Berlin Heidelberg, 2009.

[17] S. R. Chhatpar and M. S. Branicky. Search strategies for peg-in-hole assemblies with position uncertainty. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 1465–1470 vol.3, 2001.

[18] S. R. Chhatpar and M. S. Branicky. Localization for robotic assemblies using probing and particle filtering. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics.*, pages 1379–1384, 2005.

[19] N. Dantam, I. Essa, and M. Stilman. Linguistic transfer of human assembly tasks to robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 237–242, 2012.

[20] J. De Schutter, H. Bruyninckx, W.-H. Zhu, and M. W. Spong. *Force control: A bird's eye view*, pages 1–17. Springer Berlin Heidelberg, 1998.

[21] N. Delson and H. West. Robot programming by human demonstration: adaptation and inconsistency in constrained motion. In *IEEE International Conference on Robotics and Automation*, pages 30–36, 1996.

[22] F. Dimeas and N. Aspragathos. Fuzzy learning variable admittance control for human-robot cooperation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4770–4775, 2014.

[23] E. Drumwright, V. Ng-Thow-Hing, and M. Matari. Toward a vocabulary of primitive task programs for humanoid robots. In *International Conference on Development and Learning*, 2006.

[24] F. Ebrahimi, M. Mikaeili, E. Estrada, and H. Nazeran. Automatic sleep stage classification based on eeg signals by using neural networks and wavelet packet coefficients. In *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1151–1154, 2008.

[25] T.-H. Eng, Z.-K. Ling, W. Olson, and C. McLean. Feature-based assembly modeling and sequence generation. *Computers & Industrial Engineering*, 36(1):17 – 33, 1999.

[26] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent. The difficulty of training deep architectures and the effect of unsupervised pre-training. In *International Conference on artificial intelligence and statistics*, pages 153–160, 2009.

[27] D. Ewert, D. Schilberg, and S. Jeschke. Selfoptimized assembly planning for a ros based robot cell. In *5th International Conference on Intelligent Robotics and Applications*, pages 696–705, 2012.

[28] M. Faber, J. Bützler, and C. M. Schlick. Human-robot cooperation in future production systems: Analysis of requirements for designing an ergonomic work system. *Procedia Manufacturing*, 3:510 – 517, 2015. 6th International Conference on Applied Human Factors and Ergonomics and the Affiliated Conferences.

[29] K. Feldmann, V. Schöppner, and G. Spur. *Handbuch Fügen, Handhaben und Montieren*. Carl Hanser Verlag, 2013.

[30] C. Fellmann, D. Kashi, and J. Burgner-Kahrs. Evaluation of input devices for teleoperation of concentric tube continuum robots for surgical tasks. In *SPIE Medical Imaging*. International Society for Optics and Photonics, 2015.

[31] K. Fischer, F. Kirstein, L. C. Jensen, N. Krüger, K. Kukliński, M. V. aus der Wieschen, and T. R. Savarimuthu. A comparison of types of robot control for programming by demonstration. In *11th ACM/IEEE International Conference on Human-Robot Interaction*, pages 213–220, 2016.

[32] L. Gao, J. Yuan, Z. Han, S. Wang, and N. Wang. A friction model with velocity, temperature and load torque effects for collaborative industrial robot joints. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3027–3032, 2017.

[33] S. Gopinathan, S. Ötting, and J. Steil. A user study on personalized adaptive stiffness control modes for human-robot interaction. In *26th IEEE International Symposium on Robot and Human Interactive Communication*, pages 831–837, 2017.

[34] F. E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[35] E. B. Hamidullah and M. A. Irfan. Assembly features: Definition, classification, and instantiation. In *International Conference on Emerging Technologies*, pages 617–623, 2006.

[36] G.-S. Hu, F.-F. Zhu, and Z. Ren. Power quality disturbance identification using wavelet packet energy entropy and weighted support vector machines. *Expert Systems with Applications*, 35:143 – 149, 2008.

[37] T. Inoue, G. D. Magistris, A. Munawar, T. Yokoya, and R. Tachibana. Deep reinforcement learning for high precision assembly tasks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 819–825, 2017.

[38] K. Johanastrom and C. C. de Wit. Revisiting the lugre friction model. *IEEE Control Systems*, 28(6):101–114, 2008.

[39] L. P. Kaelbling and T. Lozano-Pérez. Hierarchical task and motion planning in the now. In *IEEE International Conference on Robotics and Automation*, pages 1470–1477, 2011.

[40] S. Karam and R. Teti. Wavelet transform feature extraction for chip form recognition during carbon steel turning. *Procedia CIRP*, 12:97 – 102, 2013. 8th International Conference on Intelligent Computation in Manufacturing Engineering.

[41] N. Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Applied and Computational Harmonic Analysis*, 10(3):234 – 253, 2001.

[42] P. Kormushev, S. Calinon, and D. G. Caldwell. Imitation Learning of Positional and Force Skills Demonstrated via Kinesthetic Teaching and Haptic Input. *Advanced Robotics*, 25(5):581–603, 2011.

[43] W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.

[44] J. Krüger, T. Lien, and A. Verl. Cooperation of human and machines in assembly lines. *CIRP Annals - Manufacturing Technology*, 58(2):628 – 646, 2009.

[45] Kuka AG. *KUKA LWR. User-friendly, sensitive and flexible.*, 7 2012.

[46] M. Kyrarini, M. A. Haseeb, D. Ristić-Durrant, and A. Gräser. Robot learning of industrial assembly task via human demonstrations. *Autonomous Robots*, 2018.

[47] J. S. Laursen, U. P. Schultz, and L. P. Ellekilde. Automatic error recovery in robot assembly operations using reverse execution. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1785–1792, 2015.

[48] S. H. Lee, I. H. Suh, S. Calinon, and R. Johansson. Learning basis skills by autonomous segmentation of humanoid motion trajectories. In *12th IEEE-RAS International Conference on Humanoid Robots*, pages 112–119, 2012.

[49] M. Lehto and S. Landry. *Introduction to Human Factors and Ergonomics for Engineers, Second Edition*. Human Factors and Ergonomics. Taylor & Francis, 2012.

[50] X. Li, R. Li, H. Qiao, C. Ma, and L. Li. Human-inspired compliant strategy for peg-in-hole assembly using enviromental constraint and coarse force information. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[51] J. McAuley, J. Rothwell, and C. Marsden. Frequency peaks of tremor, muscle vibration and electromyographic activity at 10 hz, 20 hz and 40 hz during human finger muscle contraction may reflect rhythmicities of central neural firing. *Experimental Brain Research*, 114(3):525–541, 1997.

[52] S. Michieletto, N. Chessa, and E. Menegatti. Learning how to approach industrial robot tasks from natural demonstrations. In *IEEE Workshop on Advanced Robotics and its Social Impacts*, pages 255–260, 2013.

[53] A. K. Mishra, L. Peña-Castillo, and O. Meruvia-Pastor. Evaluation of an inverse-kinematics depth-sensing controller for operation of a simulated robotic arm. In *Part of HCI International*, 2016.

[54] Y. Mollard, T. Munzer, A. Baisero, M. Toussaint, and M. Lopes. Robot Programming from Demonstration, Feedback and Transfer. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[55] M. Mühlig, M. Gienger, J. J. Steil, and C. Goerick. Automatic selection of task spaces for imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4996–5002, 2009.

[56] T. Munzer, M. Toussaint, and M. Lopes. Preference learning on the execution of collaborative human-robot tasks. In *IEEE International Conference on Robotics and Automation*, 2017.

[57] A. Muxfeldt, S. Gopinathan, T. Coenders, and J. Steil. A user study on human-robot-interactive recovery for industrial assembly problems. In *26th IEEE International Symposium on Robot and Human Interactive Communication*, pages 824–830, 2017.

[58] A. Muxfeldt, J.-H. Kluth, and D. Kubus. Kinesthetic teaching in assembly operations – a user study. In *4th International Conference Simulation, Modeling, and Programming for Autonomous Robots*, 2014.

[59] A. Muxfeldt and D. Kubus. Hierarchical decomposition of industrial assembly tasks. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation*, pages 1–8, 2016.

[60] A. Muxfeldt, D. Kubus, and F. M. Wahl. Developing new application fields for industrial robots - four examples for academia-industry collaboration. In *IEEE 20th Conference on Emerging Technologies Factory Automation*, pages 1–7, 2015.

[61] A. Muxfeldt and J. Steil. Fusion of human demonstrations for automatic recovery during industrial assembly. In *14th IEEE International Conference on Automation Science and Engineering*, 2018.

[62] A. Muxfeldt and J. Steil. Recovering from assembly errors by exploiting human demonstrations. In *51st CIRP Conference on Manufacturing Systems*, 2018.

[63] R. F. O'Connor, C. Baber, M. Musri, and H. Ekerol. Identification, classification and management of errors in automated component assembly tasks. *International Journal of Production Research*, 31(8):1853–1863, 1993.

[64] D. R. Olsen and M. A. Goodrich. Metrics for evaluating human-robot interactions. In *Proceedings of PERMIS*, page 4, 2003.

[65] L. Pais and A. Billard. Tactile interface user-friendliness evaluated in the context of robot programming by demonstration. HRI Workshop on Advances in tactile sensing and touch based human-robot interaction, 2012.

[66] M. R. Pedersen, L. Nalpantidis, R. S. Andersen, C. Schou, S. Bøgh, V. Krüger, and O. Madsen. Robot skills for manufacturing: From concept to industrial deployment. *Robotics and Computer-Integrated Manufacturing*, 37:282 – 291, 2016.

[67] I. Ranatunga, S. Cremer, D. O. Popa, and F. L. Lewis. Intent aware adaptive admittance control for physical human-robot interaction. In *IEEE International Conference on Robotics and Automation*, pages 5635–5640, 2015.

[68] E. Roberge and V. Duchaine. Detecting insertion tasks using convolutional neural networks during robot teaching-by-demonstration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3210–3216, 2017.

[69] P. Rouanet, J. Bechu, and P. Y. Oudeyer. A comparison of three interfaces using handheld devices to intuitively drive and show objects to a social robot: the impact of underlying metaphors. In *18th IEEE International Symposium on Robot and Human Interactive Communication*, pages 1066–1072, 2009.

[70] L. Roveda, F. Vicentini, N. Pedrocchi, F. Braghin, and L. M. Tosatti. Impedance shaping controller for robotic applications involving interacting compliant environments and compliant robot bases. In *IEEE International Conference on Robotics and Automation*, pages 2066–2071, 2015.

[71] J. Sabsch, M. Hanses, S. Zug, and N. Elkmann. Towards improving the absolute accuracy of lightweight robots by nonparametric calibration. In *IEEE 22st International Conference on Emerging Technologies and Factory Automation*, 2017.

[72] G. Salvendy. *Handbook of Industrial Engineering*. Wiley, 1992.

[73] K. Sambhoos, B. Koc, and R. Nagi. Extracting assembly mating graphs for assembly variant design. *Journal of Computing and Information Science in Engineering*, 9(3):1–9, 2009.

[74] L. Schmidt, J. Hegenberg, and L. Cramar. User studies on teleoperation of robots for plant inspection. *Industrial Robot: An International Journal*, 41(1):6–14, 2014.

[75] C. Schou, J. Damgaard, S. Bogh, and O. Madsen. Human-robot interface for instructing industrial tasks using kinesthetic teaching. In *44th International Symposium on Robotics*, pages 1–6, 2013.

[76] J. D. Schutter and H. V. Brussel. Compliant robot motion ii. a control approach based on external control loops. *The International Journal of Robotics Research*, 7(4):18–33, 1988.

[77] I. W. Selesnick, R. G. Baraniuk, and N. C. Kingsbury. The dual-tree complex wavelet transform. *IEEE Signal Processing Magazine*, 22(6):123–151, 2005.

[78] J. J. Steil and G. W. Maier. Robots in the digitalized workplace. *The Wiley Blackwell handbook of the psychology of the internet at work*, pages 401–422, 2017.

[79] J. T. C. Tan, F. Duan, R. Kato, T. Arai, and E. Hall. *Collaboration planning by task analysis in human-robot collaborative manufacturing system*. INTECH Open Access Publisher, 2010.

[80] T. Tang, H. C. Lin, Y. Zhao, W. Chen, and M. Tomizuka. Autonomous alignment of peg and hole by force/torque measurement for robotic assembly. In *IEEE International Conference on Automation Science and Engineering*, pages 162–167, 2016.

[81] U. Thomas and F. M. Wahl. *Assembly Planning and Task Planning — Two Prerequisites for Automated Robot Programming*, pages 333–354. Springer, 2011.

[82] M. Uyar, S. Yildirim, and M. T. Gencoglu. An effective wavelet-based feature extraction method for classification of power quality disturbance signals. *Electric Power Systems Research*, 78(10):1747–1755, 2008.

[83] D. Vogt, S. Stepputtis, S. Grehl, B. Jung, and H. B. Amor. A system for learning continuous human-robot interactions from human-human demonstrations. In *IEEE International Conference on Robotics and Automation*, pages 2882–2889, 2017.

[84] I. Weidauer, D. Kubus, and F. Wahl. A hierarchical extension of manipulation primitives and its integration into a robot control architecture. In *IEEE International Conference on Robotics and Automation*, pages 5401–5407, 2014.

[85] B. H. Wixom and P. A. Todd. A theoretical integration of user satisfaction and technology acceptance. *Information systems research*, 16(1):85–102, 2005.

[86] S. Wrede, O. Beyer, C. Dreyer, M. Wojtynek, and J. Steil. Vertical integration and service orchestration for modular production systems using business process models. *Procedia Technology*, 26:259–266, 2016.

[87] S. Wrede, C. Emmerich, R. Grünberg, A. Nordmann, A. Swadzba, and J. Steil. A user study on kinesthetic teaching of redundant robots in task and configuration space. *Journal of Human-Robot Interaction*, 2(1):56–81, 2013.

[88] L. D. Xu, C. Wang, Z. Bi, and J. Yu. Autoassem: An automated assembly planning system for complex products. *IEEE Transactions on Industrial Informatics*, 8(3):669–678, 2012.